

Fernando Kentaro Inaba

**Generalização do Extreme Learning Machine
Regularizado para Problemas de Múltiplas
Saídas**

Vitória

2018

Fernando Kentaro Inaba

Generalização do Extreme Learning Machine Regularizado para Problemas de Múltiplas Saídas

Tese apresentada ao programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo como requisito parcial para a obtenção do grau de Doutor em Engenharia Elétrica.

Universidade Federal do Espírito Santo – UFES

Programa de Pós-Graduação em Engenharia Elétrica – PPGEE

Laboratório de Computadores e Sistemas Neurais – LabCISNE

Orientador: Prof. Dr. Evandro Ottoni Teatini Salles

Vitória

2018

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Setorial Tecnológica,
Universidade Federal do Espírito Santo, ES, Brasil)
Sandra Mara Borges Campos – CRB-6 ES-000593/O

I35g Inaba, Fernando Kentaro, 1986-
Generalização do extreme learning machine regularizado
para problemas de múltiplas saídas / Fernando Kentaro Inaba. –
2018.

113 p. : il.

Orientador: Evandro Ottoni Teatini Salles.

Tese (Doutorado em Engenharia Elétrica) – Universidade
Federal do Espírito Santo, Centro Tecnológico.

1. Redes neurais. 2. Otimização. 3. Reconhecimento de
padrões. 4. Aprendizado do computador. 5. Inteligência
computacional. 6. Máquinas de aprendizado extremo. I. Salles,
Evandro Ottoni Teatini. II. Universidade Federal do Espírito
Santo. Centro Tecnológico. III. Título.

CDU: 621.3

Fernando Kentaro Inaba

Generalização do Extreme Learning Machine Regularizado para Problemas de Múltiplas Saídas


Tese apresentada ao programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo como requisito parcial para a obtenção do grau de Doutor em Engenharia Elétrica.

Trabalho aprovado. Vitória, 28 de fevereiro de 2018:



**Prof. Dr. Evandro Ottoni Teatini
Salles**

Universidade Federal do Espírito Santo
Orientador

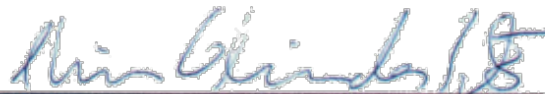


**Prof. Dr. André Carlos Ponce de Leon
Ferreira de Carvalho**

Universidade de São Paulo



Prof. Dr. Klaus Fabian Côco
Universidade Federal do Espírito Santo



Prof. Dr. Thiago Oliveira dos Santos
Universidade Federal do Espírito Santo



Prof. Dr. Thomas Walter Rauber
Universidade Federal do Espírito Santo

Vitória
2018

Agradecimentos

Gostaria de agradecer a todos que me apoiaram, de forma direta ou indireta, nos últimos anos. Não poderia deixar de citar algumas pessoas e instituições sem as quais esse trabalho seria inviabilizado. Muito obrigado...

Ao meu orientador, professor Evandro Ottoni Teatini Salles, pela paciência, ensinamentos e tempo disposto nas reuniões que moldaram o trabalho. Pelas inúmeras longas conversas, muitas vezes não relacionadas com a Tese. Pela confiança depositada em mim e pela atenção prestada ao longo dessa jornada.

Ao CNPq e CAPES pelo apoio financeiro concedido e ao Programa de Pós-Graduação em Engenharia Elétrica (PPGEE) da UFES pela oportunidade do doutorado. Aos colegas estudantes da Pós-graduação em Engenharia Elétrica pelos momentos de estudo e descontração, em especial aos colegas do laboratório CISNE. Aos amigos e familiares pelo apoio, incentivo e suporte.

Aos professores Moisés Ribeiro, Marcia Paiva e Maria José Pontes do programa de Pós-graduação em Engenharia Elétrica pela oportunidade de participar do projeto que me permitiu realizar um estágio no *Groupe d'études et de recherche en analyse des décisions* (GERAD). Aos professores Gilles Caporossi e Sylvain Perron por terem me recebido no GERD & HEC Montréal, pelos ensinamentos e atenção prestada durante meu estágio no Canadá. Ao professor Mário Sarcinelli Filho do programa de Pós-graduação em Engenharia Elétrica, por me emprestar o computador que utilizei para realizar os experimentos desta Tese. Esse permitiu uma drástica redução no tempo dos experimentos da Tese, que foi essencial para que eu conseguisse explorar todo conteúdo exposto nesta Tese.

À Marina Favrim, pela parceria, companheirismo, amor e compreensão nos últimos anos. Por conseguir, muitas vezes a quilômetros de distância, trazer a leveza e equilíbrio que precisei em diversos momentos dessa caminhada. Pela paciência constante e por me mostrar que a realidade é de uma beleza simples e sem frescuras.

Por último, mas de forma alguma menos importante, agradeço aos meus pais. Atribuo todo e qualquer sucesso profissional e pessoal que tenho ou venha a ter à eles. Trabalharam muito para que eu pudesse ter o melhor possível, mesmo quando exigia sacrifícios da parte deles. Pelo amor incondicional e compreensão durante os momentos de falta, meus sinceros agradecimentos.

Resumo

Extreme Learning Machine (ELM) recentemente teve sua popularidade aumentada e com isso tem sido aplicada com sucesso em diversas áreas do conhecimento. No estado da arte, variações usando regularização se tornaram comuns na área de ELM. A regularização mais comum é a norma ℓ_2 , que melhora a generalização, mas resulta em uma rede densa. Regularização baseada em *elastic net* também foi proposta, mas geralmente restrita a problemas de regressão de uma saída e classificação binária. Este trabalho tem por objetivo propor uma generalização do ELM regularizado para problemas de classificação multiclasse e regressão *multitarget*. Nesse sentido, observou-se que o uso da norma $\ell_{2,1}$ e de Frobenius proporcionaram uma generalização adequada. Como consequência, foi possível demonstrar que as técnicas R-ELM e OR-ELM, consagradas na literatura, são casos particulares dos métodos propostos nesta Tese, denominados de GR-ELM e GOR-ELM, respectivamente. Além disso, outra proposta deste trabalho é trazer um algoritmo alternativo para o GR-ELM, o DGR-ELM, para tratar dados que são naturalmente distribuídos. O *alternating direction method of multipliers* (ADMM) é usado para resolver os problemas de otimização resultantes. *Message passing interface* (MPI) no estilo de programação *single program, multiple data* (SPMD) é escolhido para implementar o DGR-ELM. Vários experimentos são conduzidos para avaliar os métodos propostos. Os experimentos realizados mostram que o GR-ELM, DGR-ELM e GOR-ELM possuem desempenho similar quando comparados ao R-ELM e OR-ELM, embora, em geral, uma estrutura mais compacta é obtida na rede resultante.

Palavras-chave: norma $\ell_{2,1}$. regularização. *extreme learning machine*. classificação multiclasse. *multitarget regression*. robustez à *outliers*. *alternating direction method of multipliers*.

Abstract

Extreme Learning Machine (ELM) has recently gained popularity and has been successfully applied to a wide range of applications. Variants using regularization are now a common practice in the state of the art in the ELM field. The most commonly used regularization is the ℓ_2 norm, which improves generalization but results in a dense network. Regularization based on the elastic net has also been proposed but mainly applied to regression and binary classification problems. In this thesis, it is proposed a generalization of regularized ELM (R-ELM) for multiclass classification and multitarget regression problems. The use of $\ell_{2,1}$ and Frobenius norm provided an appropriate generalization. Consequently, it was possible to show that R-ELM and OR-ELM are a particular case of the methods proposed in this thesis, termed GR-ELM and GOR-ELM, respectively. Furthermore, another method proposed in this thesis is the DGR-ELM, which is an alternative method of GR-ELM for dealing with data that are naturally distributed. The alternating direction method of multipliers (ADMM) is the algorithm used to solve the resulting optimization problems. Message Passing Interface (MPI) in a Single Program, Multiple Data (SPMD) programming style is chosen for implementing DGR-ELM. Extensive experiments are conducted to evaluate the proposed method. The conducted experiments show that GR-ELM, DGR-ELM, and GOR-ELM have similar training and testing performance when compared to R-ELM and OR-ELM, although usually inferior testing time is obtained with our method due to the compactness of the resulting network.

Keywords: $\ell_{2,1}$ norm. regularization. extreme learning machine. multiclass classification. multitarget regression. outlier robust. alternating direction method of multipliers.

Lista de ilustrações

Figura 1 – Comparação entre o quadrado da norma ℓ_2 e a norma ℓ_1 em função de uma amostra de erro, ϵ	29
Figura 2 – Ilustração de uma arquitetura SFLN com n entradas, \tilde{N} nós ocultos, e m saídas.	34
Figura 3 – Eliminação de neurônios que possuem todos os pesos associados ao mesmo iguais a zero, pela regularização com a norma $\ell_{2,1}$ em uma arquitetura típica para GR-ELM com n entradas e m saídas.	39
Figura 4 – Comparação entre os métodos em relação ao aRRMSE de treino e teste usando o teste de Nemenyi com diferentes proporções de <i>outliers</i> no banco de dados de treino.	72
Figura A.1 – Dados usados para exemplificar o efeito da regularização.	91
Figura A.2 – Aproximação obtida por rede neural treinada pelo ELM sem regularização e com (a) 1, (b) 2, (c) 5, (d) 10, (e) 15 e (f) 20 neurônios.	93
Figura A.3 – Aproximação obtida por rede neural treinada pelo ELM com regularização <i>ridge</i> e parâmetro C igual a (a) 0.01, (b) 1, (c) 10, (d) 100, (e) 10000 e (f) 100000.	94
Figura A.4 – Aproximação obtida por rede neural treinada pelo ELM com regularização ℓ_1 e parâmetro λ igual a (a) 1, (b) 0.1, (c) 0.01, (d) 0.001, (e) 0.0001 e (f) 0.00001.	97
Figura B.1 – <i>Soft-thresholding operator</i> para $\kappa = 1$	99
Figura G.1 – Comparação de quatro métodos entre si através com o teste de Nemenyi. Grupos de métodos que não são estatisticamente diferentes (a um nível de significância de 0.05) estão conectados por uma linha expessa. . . .	112

Lista de tabelas

Tabela 1	– Informações sobre o banco de dados de classificação binária.	49
Tabela 2	– Informação sobre o banco de dados de classificação multiclasse.	49
Tabela 3	– Acurácia média com os respectivos desvios padrões e tempo de execução para os bancos de dados de classificação binária ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	52
Tabela 4	– Acurácia média com os respectivos desvios padrões e tempos de execução para bancos de dados de classificação multiclasse ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	53
Tabela 5	– Especificações de parâmetros dos bancos de dados binários e o número médio de neurônios ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	54
Tabela 6	– Especificações de parâmetros para os bancos de dados multiclasses e número médio de neurônios ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	54
Tabela 7	– Acurácia média com os respectivos desvios padrões para treino e teste nos problemas de classificação binária usando o DGR-ELM com 1, 2 e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	56
Tabela 8	– Acurácia média com os respectivos desvios padrões nos problemas de classificação multiclasse usando o DGR-ELM com 1, 2 e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	56
Tabela 9	– Tempo médio para problemas de classificação binária usando o DGR-ELM com 1, 2, e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	57
Tabela 10	– Tempo médio para problemas de classificação multiclasse usando o DGR-ELM com 1, 2, e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	57
Tabela 11	– Informações sobre o banco de dados de MTR.	59
Tabela 12	– Informações sobre o banco de dados de regressão de uma saída.	59
Tabela 13	– aRRMSE de Treino e Teste para os bancos de dados MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e sem <i>outliers</i>).	63
Tabela 14	– aRRMSE Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e 10% do conjunto de treino com <i>outliers</i>).	64
Tabela 15	– aRRMSE de Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e 20% do conjunto de treino com <i>outliers</i>).	65
Tabela 16	– aRRMSE de Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e 30% do conjunto de treino com <i>outliers</i>).	66
Tabela 17	– aRRMSE de Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e 40% do conjunto de treino com <i>outliers</i>).	67
Tabela 18	– RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e sem <i>outlier</i>).	68
Tabela 19	– RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 10% do conjunto de treino com <i>outliers</i>).	68

Tabela 20 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 20% do conjunto de treino com <i>outliers</i>).	69
Tabela 21 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 30% do conjunto de treino com <i>outliers</i>).	69
Tabela 22 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 40% do conjunto de treino com <i>outliers</i>).	70
Tabela 23 – Especificações dos parâmetros para os bancos de dados de MTR ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	71
Tabela 24 – Especificações dos parâmetros para os bancos de dados de MTR ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	73
Tabela 25 – Média do número de neurônios ao final do treinamento para cada banco de dados de MTR nos métodos GOR-ELM e GR-ELM.	74
Tabela 26 – Média do número de neurônios ao final do treinamento para cada banco de dados de regressão de uma saída nos métodos GOR-ELM e GR-ELM.	75
Tabela 27 – Média dos tempos médios de treino e teste para os bancos de dados de MTR ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	76
Tabela 28 – Média dos tempos médios de treino e teste para os bancos de dados de regressão de uma saída ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).	76
Tabela A.1–Norma ℓ_1 e ℓ_2 dos pesos de saída, β , para 1, 2, 5, 10, 15, e 20 neurônios.	92
Tabela A.2–Norma ℓ_1 e ℓ_2 dos pesos de saída, β , para C igual a 0.01, 1, 10, 100, 10000, e 100000.	95
Tabela A.3–Norma ℓ_1 e ℓ_2 dos pesos de saída, β , para λ igual a 1, 0.1, 0.01, 0.001, 0.00001, e 0.000001.	95

Lista de abreviaturas e siglas

aRRMSE	average Relative Root Mean Square Error
ADMM	Alternating Direction Method of Multipliers
ALM	Augmented Lagrange Multiplier
DAM	Dual Ascent Method
DC	Distância Crítica
DD	Dual Decomposition
DGR-ELM	Distributed Generalized Regularized Extreme Learning Machine
GSL	GNU Scientific Library
GOR-ELM	Generalized Outlier Robust Extreme Learning Machine
GPU	Graphics Processing Unit
GR-ELM	Generalized Regularized Extreme Learning Machine
ELM	Extreme Learning Machine
LAPACK	Linear Algebra PACKage
lasso	least absolute shrinkage and selection operator
MM	Multiplier Method
MPI	Message Passing Interface
MTR	Multi-target Regression
OR-ELM	Outlier Robust Extreme Learning Machine
PR-ELM	Parallel Regularized Extreme Learning Machine
R-ELM	Regularized Extreme Learning Machine
RRMSE	Relative Root Mean Square Error
SPMD	Single Program Multiple Data
SLFN	Single Layer Feedforward Neural Network

SVM Support Vector Machine

UWR-ELM Unweighted Regularized Extreme Learning Machin

WR-ELM Weighted Regularized Extreme Learning Machin

Sumário

1	INTRODUÇÃO	19
1.1	Proposta	22
1.2	Contribuições	23
1.3	Organização do Trabalho	23
1.4	Notações utilizadas ao longo do texto	24
1.5	Métricas de Avaliação	24
2	REFERENCIAL TEÓRICO	27
2.1	Esparsidade	27
2.2	Norma ℓ_1 e robustez à <i>outliers</i>	28
2.3	<i>Dual Ascent</i>	30
2.4	<i>Dual Decomposition</i>	30
2.5	Lagrangiano Aumentado e Método dos Multiplicadores	31
2.6	ADMM	32
2.7	<i>Global Variable Consensus ADMM</i>	32
2.8	<i>Extreme Learning Machine</i>	33
2.9	<i>Regularized Extreme Learning Machine</i>	34
2.10	<i>Outlier Robust Extreme Learning Machine</i>	36
3	MÉTODO PROPOSTO	37
3.1	<i>Generalized Regularized Extreme Learning Machine</i>	38
3.2	<i>Distributed GR-ELM</i>	41
3.3	<i>Generalized Outlier Robust Extreme Learning Machine</i>	43
3.4	Nota computacional	46
3.5	Critério de Parada	47
3.6	Generalizações	48
4	EXPERIMENTOS DE CLASSIFICAÇÃO MULTICLASSE	49
4.1	Especificações dos Parâmetros	50
4.2	Avaliação do GR-ELM	50
4.3	Avaliação do DGR-ELM	55
5	EXPERIMENTOS DE REGRESSÃO COM OUTLIERS	59
5.1	Especificação dos Parâmetros	60
5.2	Avaliação do GOR-ELM	61
6	CONCLUSÃO	77

REFERÊNCIAS	81
 APÊNDICES	 89
APÊNDICE A – CONEXÕES ENTRE REGULARIZAÇÃO, <i>OVER-FITTING</i> E ESPARSIDADE EM ELM	91
APÊNDICE B – <i>SOFT-THRESHOLDING OPERATOR</i>	99
APÊNDICE C – DERIVAÇÕES MATEMÁTICAS DAS ATUALIZAÇÕES DO ADMM	101
APÊNDICE D – FATORAÇÃO DE CHOLESKY	105
APÊNDICE E – SUBSTITUIÇÃO <i>FORWARD</i> E <i>BACKWARD</i> . .	107
APÊNDICE F – NORMAS	109
APÊNDICE G – TESTES ESTATÍSTICOS	111

1 Introdução

Uma das primeiras publicações, relacionadas ao que se conhece hoje por redes neurais, data de 1943, quando McCulloch e Pitts propuseram um modelo matemático inspirado nos neurônios biológicos para representar funções lógicas, que resultou no primeiro conceito de neurônio artificial.

Em 1958, Rosenblatt propôs uma estrutura de rede neural chamada *perceptron*, além de definir um algoritmo de aprendizagem para essa rede. Nos anos 70 as pesquisas na área ficaram adormecidas, em grande parte devido ao trabalho de Minsky e Papert (1969) que argumentaram limitações teóricas do *perceptron*. Apesar de Werbos, em 1974, ter proposto um modelo matemático cuja forma de treinamento guarda grande semelhança com redes neurais de múltiplas camadas com retropropagação de erro, foi somente em 1986, que Rumelhart, Hinton e Williams, de forma independente de Werbos, formalizaram um algoritmo para aprendizagem, chamado de *error backpropagation*, despertando a atenção da comunidade acadêmica. Nesse trabalho, os autores mostraram que uma estrutura baseada no *perceptron* em múltiplas camadas era capaz de ser treinada e assim resolver problemas complexos, ao contrário do que foi argumentado por Minsky e Papert (1969). Desde então, as redes neurais vêm sendo aplicadas nas mais diversas áreas do conhecimento, com diferentes estruturas e paradigmas, possuindo uma sólida e extensa teoria.

Redes neurais *feedforward* têm sido muito estudadas e amplamente usadas desde a introdução do algoritmo *backpropagation* por Rumelhart, Hinton e Williams (1986). Haykin (1999), de forma genérica, define uma rede neural como sendo um processador maciçamente paralelo distribuído, constituído de unidades de processamento simples (neurônios), que encontram-se interconectadas, através de pesos que possuem a capacidade de armazenar conhecimento por experiência adquirida e torná-lo disponível para o uso.

Dentre as arquiteturas de redes neurais existentes, a presente Tese trabalha com as *single layer feedforward neural network* (SLFN), que possui apenas uma camada (oculta) entre as entradas e saídas. O termo *feedforward* refere-se ao fato das redes SLFN propagarem para frente o fluxo de informação de forma unidirecional, isto é, não há realimentação na rede.

O algoritmo *backpropagation* tradicional é essencialmente um método de descida gradiente de primeira ordem para otimização de parâmetros, que é conhecido por ter convergência lenta e problemas com mínimo local (HUANG et al., 2015). Diversas alternativas foram propostas para melhorar a eficiência ou otimalidade no treinamento das redes neurais *feedforward*, tais como métodos de otimização de segunda ordem, de seleção de modelos ou de otimização global (HAGAN; MENHAJ, 1994; WILAMOWSKI; YU, 2010;

CHEN; COWAN; GRANT, 1991; LI; PENG; IRWIN, 2005; BRANKE, 1995; YAO, 1993; HUANG et al., 2015). Embora esses métodos resultem em um treinamento mais rápido, ou em uma melhor generalização quando comparado com o algoritmo *backpropagation*, a maioria desses métodos não garantem a solução ótima global (HUANG et al., 2015).

Huang (2003), Tamura e Tateishi (1997) e Huang, Zhu e Siew (2006) mostraram que uma SLFN, com \widetilde{N} neurônios na camada oculta e pesos de entrada escolhidos aleatoriamente podem aprender exatamente \widetilde{N} observações distintas. Baseado nisso, Huang, Zhu e Siew (2006) propuseram um treinamento de redes SLFN rápido, denominado de *Extreme Learning Machine* (ELM), enquanto obtém uma melhor generalização quando comparado com os algoritmos de treinamento tradicionais, como o *backpropagation*.

A popularidade do *Extreme Learning Machine* (ELM) proposta em (HUANG et al., 2012; HUANG; ZHU; SIEW, 2006) e suas variantes (HUANG et al., 2015; WANG; ER; HAN, 2014) aumentou recentemente e esses métodos têm sido usados com sucesso em diversas aplicações tais como em visão computacional, processamento de imagem, análise de séries temporais e aplicações médicas (HUANG et al., 2015). Nessas áreas de pesquisas, ELM demonstra uma boa generalização e geralmente possui um desempenho melhor que o tradicional *support vector machine* (SVM) (HUANG et al., 2015), ao mesmo tempo que mantém um baixo custo computacional, justificando assim o aumento de sua popularidade. A ideia principal do ELM é gerar, aleatoriamente, os pesos de entrada de uma *single hidden layer feedforward neural network* (SLFN) e então obter de forma determinística os pesos de saída (HUANG et al., 2012; WANG; ER; HAN, 2015).

Em ELM (HUANG; ZHU; SIEW, 2006), um dos parâmetros que precisa ser bem escolhido é o número de neurônios na camada oculta para obter um bom compromisso entre *underfitting/overfitting*. Para contornar isso, Deng, Zheng e Chen (2009) propuseram uma versão regularizada do ELM usando a teoria de regressão *ridge*. Embora essa abordagem resulte em uma boa generalização, a rede obtida é densa e geralmente requer mais espaço de armazenamento e tempo de processamento de novas amostras para aplicações com grandes volumes de dados (BAI et al., 2014). Além disso, nessas aplicações, o ELM e sua versão com regularização *ridge* podem sofrer com a limitação de memória e um custo computacional intenso para inversão de grandes matrizes (WANG et al., 2016).

Usando a teoria de *elastic net*, Martínez-Martínez et al. (2011) propuseram o *regularized extreme learning machine* (R-ELM) para seleção automática da arquitetura da rede neural em problemas de regressão. Para classificação binária, a extensão do R-ELM é direta. Para problemas de classificação multiclasse, abordagens usando classificadores *one-versus-rest* podem ser adotados (HUANG et al., 2012; CHOROWSKI; WANG; ZURADA, 2014). Contudo, um número grande de classes geralmente resulta em um maior custo computacional na etapa de treinamento. Embora o uso do R-ELM geralmente resulte em uma rede mais compacta, esse sofre dos mesmos problemas que o ELM com regularização

ridge para tarefas de aprendizado com volumes de dados maiores.

Recentemente, Wang et al. (2016) propuseram o *parallel regularized extreme learning machine* (PR-ELM) para melhorar a eficiência computacional do ELM quando aplicado à problemas com volumes de dados maiores. Em PR-ELM, o conjunto de dados é dividido em pequenos pedaços que são tratados em diferentes nós computacionais e evita procedimentos computacionalmente intensos de operações de multiplicação e inversão de matrizes grandes. Entretanto, PR-ELM considera apenas a regularização *ridge* e a rede neural resultante é densa e geralmente requer mais tempo na execução de testes com novas amostras.

Com o advento da era do *big data*, outro problema que se agrava é o erro de instrumentação ou humano no processo de aquisição dos dados, também conhecido como *outliers*, que são dados discrepantes das amostras regulares, que podem ocorrer nos dados de treino (ZHANG; LUO, 2015; HODGE; AUSTIN, 2004). Apesar do ELM apresentar, como comentado anteriormente, uma boa generalização (especialmente com a regularização *ridge*), o mesmo sofre na presença de *outliers* nos dados de treino comumente aparentes em aplicações reais (HORATA; CHIEWCHANWATTANA; SUNAT, 2013; ZHANG; LUO, 2015).

Horata, Chiewchanwattana e Sunat (2013) propuseram o uso da família dos M-estimadores para melhorar a robustez à *outliers*, nomeando o método de *iteratively reweighted ELM*. Contudo, sua formulação não contempla um termo regularizador nos pesos de saída. Chen et al. (2017) contornam esse problema inserindo um termo regularizador, contudo, tais métodos demandam um alto custo computacional, já que cada iteração envolve a mesma quantidade de tempo gasto pelo ELM original. Zhang e Luo (2015) propuseram o *outlier robust ELM* (OR-ELM), onde a norma ℓ_1 é usada como função de perda e a norma ℓ_2 é usada no termo de regularização dos pesos.

Como ressaltado por Zhang e Luo (2015), pouco trabalho se tem feito na melhoria da robustez à *outliers* no ELM. Os métodos citados até o momento, apesar de contemplar a melhoria da robustez à *outliers* em suas abordagens, suas aplicações são focadas em problemas que possuem apenas uma saída, ou quando consideram múltiplas saídas, não consideram estrutura alguma das mesmas. Além disso, as redes obtidas são densas e podem resultar em um uso de neurônios maior do que o necessário para se obter a mesma aproximação.

A configuração, talvez mais comumente encontrada em problemas de *machine learning*, consiste na obtenção de um método que seja capaz de prever um único atributo de saída, seja ele categórico ou numérico (AHO et al., 2012). Entretanto, em muitos problemas reais, deseja-se prever múltiplos atributos de saídas relacionadas, simultaneamente. *Multi-label classification* é um exemplo onde uma amostra de entrada pode estar associada à múltiplos *labels* em problemas de classificação, já bastante explorados na literatura (KONGSOROT; HORATA; SUNAT, 2015; ZHANG; ZHANG, 2016; SUN et al., 2016;

ZHANG; DING; ZHANG, 2016; JIANG; PAN; LI, 2017). Um problema parecido, ainda pouco explorado na literatura de ELM, mas comum em outras áreas, é o *multi-target regression* (MTR) (APPICE; DZEROSKI, 2007; AHO et al., 2012), também conhecido por regressão multivariada (BROWN; ZIDEK, 1980; BREIMAN; FRIEDMAN, 1997) ou regressão de múltiplas saídas (SIMILÄ; TIKKA, 2007), que se refere a tarefa de prever múltiplas variáveis contínuas usando um conjunto comum de variáveis (SPYROMITROS-XIOUFIS et al., 2016).

MTR recentemente reobteve grande popularidade devido à sua capacidade de aprender simultaneamente múltiplas tarefas de regressão, lidando com as correlações entre as múltiplas saídas e com aplicabilidade em diversas áreas do conhecimento (e.g. *data mining*, visão computacional e análise de imagens médicas) (ZHEN et al., 2017). Nesse novo contexto, métodos que busquem a robustez em problemas de MTR ainda foram pouco explorados.

Diante do exposto, generalizações dos métodos baseados no ELM existentes, para lidar com problemas de múltiplas saídas, como problemas de classificação multiclasse e MTR, que contemplem: as relações entre as múltiplas saídas; busca por uma rede mais compacta; possibilidade de utiliza-las de forma distribuída para tratar problemas com grandes volumes de dados; e adaptável para problemas que possuem *outliers*, pode ser de grande valia.

1.1 Proposta

Diversas variantes do ELM foram propostas na literatura. Entretanto, grande parte desses métodos se concentram em problemas que possuem uma saída apenas. Os métodos que contemplam múltiplas saídas, em geral não consideram qualquer estrutura na matriz de saída. Ao invés disso, esses métodos geralmente tratam cada saída de forma isolada, e o método é repetido para cada saída de forma independente. Ademais, grande parte das variantes do ELM não se preocupam com a compacidade da rede, i.e., obter uma rede mais compacta, exceto os métodos de poda, onde é necessário fazer diversos treinamentos da rede, para retirar e/ou aumentar o número de neurônios.

Assim, levanta-se a seguinte hipótese norteadora deste trabalho: a generalização de ELM regularizados, através de normas matriciais, contemplando a compacidade da rede, é adequada para tratar problemas de classificação multiclasse e regressão de múltiplas saídas e adaptável para problemas com grandes volumes de dados. Nesse caso, deseja-se que a rede resultante da proposta seja adequada no sentido de possuir uma competitiva acurácia ou erro, quando comparada às técnicas tradicionais, e que resulte em uma rede mais compacta, quando possível.

Portanto, esta Tese tem como objetivo geral propor uma generalização de ELM

regularizados para problemas de múltiplas saídas, com regularizações que permitam obter uma rede mais compacta, que possa facilmente ser implementada de forma distribuída.

Para atingir esse objetivo geral, os seguintes objetivos específicos são definidos:

- propor um algoritmo para solução do problema de otimização resultante da formulação da generalização proposta que deve ser facilmente adaptável para problemas distribuídos;
- a depender do problema que se deseja resolver, propor mudanças de normas matriciais na generalização proposta que devem resultar em métodos capazes de tratar diferentes problemas, sejam eles de classificação multiclasse ou regressão robusta;
- tratar de problemas de classificação multiclasse e regressão com múltiplas saídas;
- tratar o problema de rebustez à *outliers* em regressão de múltiplas saídas; e
- buscar por uma rede mais compacta, empregando-se o conceito de esparsidade, mas que, em contrapartida, a compacidade não afete a acurácia ou o erro da generalização proposta.

1.2 Contribuições

- Generalização do R-ELM e OR-ELM para problemas de múltiplas saídas através de normas matriciais;
- Uso de esparsidade como forma de tornar a rede mais compacta, sem que isso comprometa a acurácia ou erro; e
- Desenvolvimento de um método que processa dados de forma distribuída.

Como resultado desse trabalho, foi publicado o artigo “*DGR-ELM–Distributed Generalized Regularized ELM for classification*”, Neurocomputing, Volume 275, 31 Janeiro 2018, Pág. 1522-1530.

1.3 Organização do Trabalho

O trabalho está organizado como se segue:

Até o momento, o Capítulo 1 contextualizou os problemas de ELM regularizados para aplicações com múltiplas saídas, definindo a hipótese e objetivo do trabalho. Ademais, será apresentado a notação utilizada ao longo do texto além de revisar alguns conceitos importantes para o entendimento do mesmo.

O Capítulo 2 abordará os principais métodos utilizados como base para a proposta elaborada. Especificamente, uma revisão de esparsidade, ELM, R-ELM, OR-ELM, além do algoritmo de otimização adotado para resolver as funções objetivos dos métodos propostos será feita nesse capítulo.

O método proposto para tratar problemas de múltiplas saídas é definido no Capítulo 3. Além da formulação geral, esse capítulo abordará duas generalizações de métodos já bem estabelecidos na literatura, além de uma implementação distribuída para um desses métodos.

Nos Capítulos 4 e 5, testes serão apresentados utilizando a metodologia descrita no Capítulo 3. Em especial dois problemas são testados nos métodos propostos: classificação multiclasse e regressão de múltiplas saídas com *outliers*. Assim, nos problemas de classificação, avalia-se a acurácia do método proposto, e para o problema de regressão de múltiplas saídas com *outliers* o erro é avaliado. Além disso, para os dois problemas, avalia-se a quantidade de neurônios necessária para resolver os problemas, além do tempo de execução dos algoritmos resultantes das propostas.

Por fim, o Capítulo 6 apresentará as discussões gerais sobre o material apresentado nesta Tese, as vantagens e desvantagens do método bem como sugestões para trabalhos futuros.

1.4 Notações utilizadas ao longo do texto

Vetores serão representados em letras minúsculas e em negrito. A menos que se diga o contrário, os vetores representarão um vetor coluna. Matrizes serão representadas por letras maiúsculas e em negrito. Para representar uma linha ou coluna de uma matriz, utiliza-se a correspondente letra minúscula em negrito da seguinte forma: suponha uma matriz \mathbf{A} , a i -ésima linha da matriz é representado por $\mathbf{a}_{i,\cdot}$, e a j -ésima coluna por $\mathbf{a}_{\cdot,j}$. O símbolo $\mathbf{0}$ representa uma matriz onde todos os elementos são iguais a zero. $\langle \mathbf{A}, \mathbf{B} \rangle$ representa o produto interno euclidiano entre duas matrizes definido por $\langle \mathbf{A}, \mathbf{B} \rangle := \text{tr}(\mathbf{A}^T \mathbf{B})$ e $\text{tr}(\cdot)$ é o traço de uma matriz. No contexto de ELM, \tilde{N} é o número de neurônios na camada oculta, $\mathbf{H} \in \mathbb{R}^{N \times \tilde{N}}$ representará a matriz de saída dos neurônios da camada oculta, $\mathbf{X} \in \mathbb{R}^{N \times n}$ é a matriz de dados de entrada com N amostras de dimensão n , $\mathbf{T} \in \mathbb{R}^{N \times m}$ ($\mathbf{t} \in \mathbb{R}^N$) a matriz (vetor) de saída, $\mathbf{B} \in \mathbb{R}^{\tilde{N} \times m}$ ($\boldsymbol{\beta} \in \mathbb{R}^{\tilde{N}}$) é a matriz (vetor) de pesos que conecta a camada oculta à camada de saída.

1.5 Métricas de Avaliação

Para avaliar os métodos nos problemas de classificação utiliza-se a acurácia, que representa a porcentagem de amostras que foram classificadas corretamente pelo método.

Em outras palavras, representa a razão entre o número de amostras classificadas corretamente pelo número total de amostras, multiplicada por 100%. Note que a acurácia é limitada por uma faixa entre 0% a 100%. Quanto maior o valor da acurácia, melhor a eficácia do método para realizar a classificação.

Para avaliar os métodos nos problemas de regressão, utiliza-se o *average relative root mean square error* (aRRMSE) (AHO et al., 2012; SPYROMITROS-XIOUFIS et al., 2016). Considere um problema de regressão onde a variável independente (saída) é $\mathbf{t} = (t_1, t_2, \dots, t_N)^\top$. Seja $\hat{\mathbf{t}} = (\hat{t}_1, \hat{t}_2, \dots, \hat{t}_N)^\top$ a estimativa de \mathbf{t} obtida por um método qualquer. O *Relative Root Mean Squared Error* (RRMSE) é definido como (AHO et al., 2012):

$$\text{RRMSE}(\hat{\mathbf{t}}, \mathbf{t}) = \sqrt{\frac{\sum_{i=1}^N (\hat{t}_i - t_i)^2}{\sum_{i=1}^N (\bar{\mathbf{t}} - t_i)^2}}, \quad (1.1)$$

onde $\bar{\mathbf{t}} = (1/N) \sum_{i=1}^N t_i$. Assim, essa métrica representa o RMSE entre a saída e sua estimativa pelo modelo, dividido pelo RMSE entre a saída e a média da mesma. Note que em aplicações reais, não se tem a média da saída para usar como estimador. Logo, essa métrica mensura o quão melhor o método está em relação à média caso fosse possível ter o conhecimento da mesma. O RRMSE assume valores maiores ou iguais a zero, sendo igual a zero quando há estimação sem erros. Logo, quanto menor o valor do RRMSE, melhor a eficácia do método para realizar a regressão.

Para problemas de regressão com múltiplas saídas, considere $\mathbf{t}^{(\ell)} = (t_1^{(\ell)}, t_2^{(\ell)}, \dots, t_N^{(\ell)})^\top$, com $\ell = 1, 2, \dots, m$, como sendo a ℓ -ésima das m saídas do problema. Assim, o aRRMSE é definido como a média do RRMSE aplicado nas m saídas, i.e.,

$$\text{aRRMSE}(\hat{\mathbf{T}}, \mathbf{T}) = \frac{1}{m} \sum_{\ell=1}^m \text{RRMSE}(\hat{\mathbf{t}}^{(\ell)}, \mathbf{t}^{(\ell)}) = \frac{1}{m} \sum_{\ell=1}^m \sqrt{\frac{\sum_{i=1}^N (\hat{t}_i^{(\ell)} - t_i^{(\ell)})^2}{\sum_{i=1}^N (\bar{\mathbf{t}}^{(\ell)} - t_i^{(\ell)})^2}}, \quad (1.2)$$

onde $\mathbf{T} = [\mathbf{t}^{(1)} \quad \mathbf{t}^{(2)} \quad \dots \quad \mathbf{t}^{(m)}]$ é a matriz de saídas, e $\hat{\mathbf{T}} = [\hat{\mathbf{t}}^{(1)} \quad \hat{\mathbf{t}}^{(2)} \quad \dots \quad \hat{\mathbf{t}}^{(m)}]$ a matriz de saídas estimada.

Tanto para os problemas de classificação quanto para os problemas de regressão, para avaliar a complexidade de treinamento e compacidade da rede obtida utiliza-se o tempo. A unidade usada para o tempo foi segundos. Quanto maior o tempo de treinamento, maior a complexidade da técnica. Entretanto, quanto menor o tempo de teste, tem-se um indicativo de que o número de neurônios resultante do treinamento foi menor.

2 Referencial Teórico

Neste capítulo apresenta-se uma breve revisão sobre os principais assuntos tratados neste trabalho. Primeiro, uma revisão é feita sobre os principais conceitos de esparsidade, sua relação com a norma ℓ_1 e como essa norma pode ser utilizada no contexto de problemas de robustez à *outliers*. Em seguida, revisa-se o ADMM, método utilizado como base para resolver os problemas de otimização tratados nesta Tese. Para isso, os métodos precursores ao ADMM são apresentados, bem como suas limitações que levaram ao desenvolvimento do ADMM. Por fim, o ELM, e duas variações importantes do mesmo, são revistos.

No Apêndice A é feita uma discussão sobre as conexões entre regularização, *overfitting* e esparsidade, muito tratadas nesse capítulo.

2.1 Esparsidade

Na última década, esparsidade surgiu como um dos conceitos conducentes em uma variedade de aplicações em processamento de sinais (representação, extração de característica, separação de fonte e compressão, para citar algumas das aplicações) e *machine learning*. Esparsidade tornou-se teoricamente atrativa e uma propriedade prática em muitas áreas da matemática aplicada (STARCK; MURTAGH; FADILI, 2010).

Considere um vetor $\boldsymbol{\beta} \in \mathbb{R}^{\tilde{N}}$, $\boldsymbol{\beta} = [\beta_1 \ \dots \ \beta_{\tilde{N}}]^T$. Esse vetor é exatamente ou estritamente esparso se a maioria de seus elementos forem iguais a zero, isto é, se o suporte $\Lambda(\boldsymbol{\beta}) = \{1 \leq i \leq \tilde{N} \mid \beta_i \neq 0\}$ é de cardinalidade $k \ll N$ (STARCK; MURTAGH; FADILI, 2010). Portanto, um sinal é dito k -esparso quando possui no máximo k elementos não nulos.

Suponha ainda que deseja-se resolver o sistema representado por

$$\mathbf{y} = \mathbf{H}\boldsymbol{\beta}_0, \quad (2.1)$$

onde $\boldsymbol{\beta}_0 \in \mathbb{R}^{\tilde{N}}$ é um vetor esparso e desconhecido, $\mathbf{y} \in \mathbb{R}^N$ e $\mathbf{H} \in \mathbb{R}^{N \times \tilde{N}}$.

Uma tentativa natural de obtenção de $\boldsymbol{\beta}_0$ é determinar a solução do sistema linear de equações $\mathbf{y} = \mathbf{H}\boldsymbol{\beta}$. Se $N > \tilde{N}$, o sistema de equações $\mathbf{y} = \mathbf{H}\boldsymbol{\beta}$ é sobredeterminado, e geralmente a correta solução, $\boldsymbol{\beta}_0$, pode ser encontrada. Entretanto, se $N < \tilde{N}$, o sistema é subdeterminado e, portanto, a solução não é única. De maneira a contornar isto, é comum a escolha da solução que possui a menor norma ℓ_2 .

$$(\ell_2) : \quad \hat{\boldsymbol{\beta}}_2 = \arg \min \|\boldsymbol{\beta}\|_2 \quad \text{sujeito a} \quad \mathbf{H}\boldsymbol{\beta} = \mathbf{y}. \quad (2.2)$$

Este problema pode ser facilmente resolvido utilizando a pseudo-inversa de \mathbf{H} , contudo a solução $\hat{\boldsymbol{\beta}}_2$ é geralmente densa, e portanto possui vários elementos diferentes de zero.

Como mencionado anteriormente, deseja-se que a solução seja esparsa e por isso a solução do problema de otimização

$$(\ell_0) : \quad \hat{\beta}_0 = \arg \min \|\beta\|_0 \quad \text{sujeito a} \quad \mathbf{H}\beta = \mathbf{y}, \quad (2.3)$$

onde $\|\cdot\|_0$ representa o número de elementos diferentes de zero¹ do vetor é mais apropriada. Contudo, este problema é *NP*-completo (AMALDI; KANN, 1998), tornando-se computacionalmente inviável.

Se a solução β_0 é suficientemente esparsa, a solução para o problema dado pela Equação 2.3 é igual a solução do seguinte problema de otimização (CHEN; DONOHO; SAUNDERS, 1998):

$$(\ell_1) : \quad \hat{\beta}_1 = \arg \min \|\beta\|_1 \quad \text{sujeito a} \quad \mathbf{H}\beta = \mathbf{y}. \quad (2.4)$$

O problema da Equação 2.4 pode ser resolvido em tempo polinomial utilizando métodos de programação linear (CHEN; DONOHO; SAUNDERS, 1998; WRIGHT et al., 2009).

O modelo da Equação 2.1 pode ser modificado, quando a igualdade não é verdadeira (como no caso de presença de ruído), para considerar explicitamente o ruído

$$\mathbf{y} = \mathbf{H}\beta_0 + \epsilon, \quad (2.5)$$

onde $\epsilon \in \mathbb{R}^N$ é o ruído com energia limitada $\|\epsilon\|_2 < \varepsilon$. A solução β_0 pode ser aproximadamente recuperada resolvendo-se o problema de minimização- ℓ_1 estável

$$(\ell_{1s}) : \quad \hat{\beta}_1 = \arg \min \|\beta\|_1 \quad \text{sujeito a} \quad \|\mathbf{H}\beta - \mathbf{y}\|_2 \leq \varepsilon. \quad (2.6)$$

A letra *s* de ℓ_{1s} refere-se à estável (do inglês, *stable*). Dificilmente a igualdade $\mathbf{H}\beta = \mathbf{y}$ da restrição do problema posto na Equação 2.4 é assegurada, especialmente na presença de ruídos. Portanto, na versão estável tolera-se um ruído de energia limitada $\|\epsilon\|_2 < \varepsilon$, para garantir que a restrição do problema seja atendida e, assim, haja convergência do algoritmo.

2.2 Norma ℓ_1 e robustez à *outliers*

Recentemente, a norma ℓ_1 tem sido aplicada para melhorar a robustez de problemas como o definido na Equação 2.5, mas agora \mathbf{y} contendo *outliers* (XU et al., 2013; ZHANG; LUO, 2015; CHEN et al., 2017). Para isso, utiliza-se a norma ℓ_1 como função perda, i.e.,

$$\hat{\beta}_r = \arg \min \|\epsilon\|_1 \quad \text{sujeito a} \quad \mathbf{y} - \mathbf{H}\beta = \epsilon, \quad (2.7)$$

¹ Comumente chamado na literatura de norma ℓ_0 , embora não seja uma norma por não atender todas as propriedades de uma norma (ver Apêndice F).

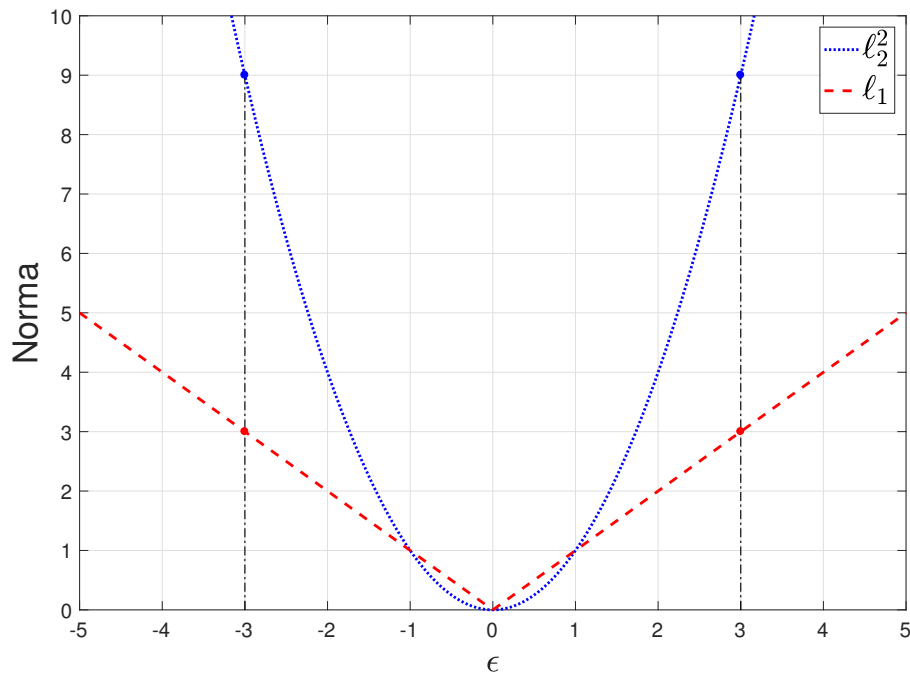
ao invés de utilizar a tradicional norma ℓ_2 , i.e.,

$$\hat{\beta}_t = \arg \min \|\mathbf{y} - \mathbf{H}\beta\|_2^2. \quad (2.8)$$

Nessa configuração, em geral, os *outliers* tendem a gerar um vetor de erro esparsos, e como comentado na seção anterior, a norma ℓ_1 é adequada nesse caso. Além disso, outro motivo para a norma ℓ_1 apresentar mais robustez à *outliers* quando comparada com a norma ℓ_2 é que no caso de *outliers*, esses dados são atípicos e por isso resultam em um erro grande. Em uma função de perda quadrática, como é o caso da Equação 2.8, o erro, que já é grande, é elevado ao quadrado e portanto o erro gerado pelos *outliers* domina a função custo. Ao usar a norma ℓ_1 , a influência dos erros referentes às amostras com *outliers* é diminuída.

A Figura 1 exibe a norma ℓ_1 e o quadrado da norma ℓ_2 em função do erro ϵ para o caso de uma amostra. Note que à medida que o erro aumenta, em módulo, a contribuição dessa amostra na função custo é linear quando se usa a norma ℓ_1 e quadrática para a norma ℓ_2 . Em especial, quando se tem um erro $\epsilon = 3$, a sua norma ℓ_2 (ao quadrado) é nove e no caso da norma ℓ_1 é três. Assim, a solução para o problema que utiliza a norma ℓ_2 irá priorizar eliminar o erro de *outlier*, fazendo com que o erro nas amostras que não possuem *outliers* tenham um erro maior.

Figura 1 – Comparação entre o quadrado da norma ℓ_2 e a norma ℓ_1 em função de uma amostra de erro, ϵ .



Fonte: Produção do próprio autor.

2.3 Dual Ascent

Considere o seguinte problema de otimização convexa com restrições de igualdade:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimizar}} && f(\mathbf{x}) \\ & \text{sujeito a} && \mathbf{Ax} = \mathbf{b}, \end{aligned} \quad (2.9)$$

onde $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$, e $f : \mathbb{R}^n \rightarrow \mathbb{R}$ é convexa. Esse problema é dito ser um problema primal (para a função primal f) e \mathbf{x} é chamado de variável primal. O Lagrangiano para o problema da Equação 2.9 é

$$L(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) + \mathbf{y}^\top (\mathbf{Ax} - \mathbf{b}). \quad (2.10)$$

A função $g(\mathbf{y}) = \inf_{\mathbf{x}} L(\mathbf{x}, \mathbf{y})$ é dita ser a função dual e o problema dual é

$$\underset{\mathbf{y}}{\text{maximizar}} \quad g(\mathbf{y}), \quad (2.11)$$

onde $\mathbf{y} \in \mathbb{R}^m$ é a variável dual. O ponto ótimo primal, \mathbf{x}^* , pode ser recuperado a partir do ponto ótimo dual, \mathbf{y}^* , como

$$\mathbf{x}^* = \underset{\mathbf{x}}{\text{minimizar}} \quad L(\mathbf{x}, \mathbf{y}^*), \quad (2.12)$$

quando há apenas uma solução para $L(\mathbf{x}, \mathbf{y}^*)$ (um exemplo de quando isso corre é quando f é estritamente convexa).

No método *dual ascent* (DAM), o problema dual é resolvido usando a subida de gradiente. Assumindo que g é diferenciável, o DAM consiste nas iterações das seguintes atualizações

$$\mathbf{x}^{k+1} := \arg \min_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}^k) \quad (2.13)$$

$$\mathbf{y}^{k+1} := \mathbf{y}^k + \alpha^k (\mathbf{Ax}^{k+1} - \mathbf{b}), \quad (2.14)$$

onde $\alpha^k > 0$ é o tamanho do passo na iteração k . Note que $\nabla g(\mathbf{y}^k) = \mathbf{Ax}^+ - \mathbf{b}$, e $\mathbf{x}^+ = \arg \min_{\mathbf{x}} L(\mathbf{x}, \mathbf{y}^k)$.

2.4 Dual Decomposition

Uma característica importante do DAM é que o método pode ser paralelizado em alguns casos (BOYD, 2010). Suponha que a função objetivo f é separável, i.e. $f(\mathbf{x}) = f_1(\mathbf{x}_1) + \dots + f_n(\mathbf{x}_N)$ e $\mathbf{x} = (\mathbf{x}_1^\top, \dots, \mathbf{x}_N^\top)^\top$, onde as variáveis $\mathbf{x}_i \in \mathbb{R}^{n_i}$ são subvetores de \mathbf{x} . A matriz \mathbf{A} pode ser particionada como $\mathbf{A} = [\mathbf{A}_1 \quad \dots \quad \mathbf{A}_N]$, de modo que $\mathbf{Ax} = \sum_{i=1}^N \mathbf{A}_i \mathbf{x}_i$, e o Lagrangiano pode ser escrito como

$$L(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^N L_i(\mathbf{x}_i, \mathbf{y}) = \sum_{i=1}^N \left(f_i(\mathbf{x}_i) + \mathbf{y}^\top \mathbf{A}_i \mathbf{x}_i - (1/N) \mathbf{y}^\top \mathbf{b} \right), \quad (2.15)$$

que também é separável em \mathbf{x} . Assim, o problema de minimização para atualização de \mathbf{x} do DAM (Equação 2.13) se separa em N problemas, que podem ser resolvidos em paralelo conforme

$$\mathbf{x}_i^{k+1} := \arg \min_{\mathbf{x}_i} L_i(\mathbf{x}_i, \mathbf{y}^k). \quad (2.16)$$

Segue que a atualização da variável dual no DAM pode ser escrita conforme

$$\mathbf{y}^{k+1} := \mathbf{y}^k + \alpha^k \left(\sum_{i=1}^N \mathbf{A}_i \mathbf{x}_i^{k+1} - \mathbf{b} \right). \quad (2.17)$$

O algoritmo composto pelas atualizações Equação 2.16 e Equação 2.17 é chamado de *dual decomposition* (DD).

2.5 Lagrangiano Aumentado e Método dos Multiplicadores

O Lagrangiano aumentado torna o DAM mais robusto, e em particular, resulta em uma convergência mais rápida e sem a necessidade de assumir convexidade estrita para f . O Lagrangiano aumentado para Equação 2.9 é

$$L_\rho(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) + \mathbf{y}^\top (\mathbf{Ax} - \mathbf{b}) + (\rho/2) \|\mathbf{Ax} - \mathbf{b}\|_2^2, \quad (2.18)$$

onde $\rho > 0$ é o parâmetro de penalização. Note que se ρ for igual a zero, tem-se o Lagrangiano padrão.

O Lagrangiano aumentado pode ser visto como o Lagrangiano padrão associado ao problema

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimizar}} && f(\mathbf{x}) + (\rho/2) \|\mathbf{Ax} - \mathbf{b}\|_2^2 \\ & \text{sujeito a} && \mathbf{Ax} = \mathbf{b}. \end{aligned} \quad (2.19)$$

Note que para qualquer solução viável \mathbf{x} , o termo adicionado na função objetivo é igual a zero. Assim, tem-se que o problema dado pela Equação 2.19 é equivalente ao problema original da Equação 2.9. É possível mostrar que o Lagrangiano aumentado é diferenciável sob condições pouco restritivas para o problema primal (BOYD, 2010). Aplicando o DAM no problema modificado (Equação 2.19) resulta no seguinte algoritmo

$$\mathbf{x}^{k+1} := \arg \min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{y}^k) \quad (2.20)$$

$$\mathbf{y}^{k+1} := \mathbf{y}^k + \rho (\mathbf{Ax}^{k+1} - \mathbf{b}), \quad (2.21)$$

que é chamado de Método dos Multiplicadores (MM), também conhecido como *augmented Lagrange multiplier* (ALM), para resolver a Equação 2.9. Note que o MM é o mesmo que o DAM, exceto pela etapa de atualização de \mathbf{x} , que utiliza o Lagrangiano aumentado, e o parâmetro de penalidade, ρ , é usado como tamanho do passo α^k . O MM converge em condições mais gerais do que o DAM, incluindo o caso de f assumir valores $+\infty$ ou não ser estritamente convexa. Essa melhora nas propriedades de convergência do MM quando

comparado ao DAM vem com um custo. Quando f é separável, o Lagrangiano aumentado L_ρ não é separável. Assim, a etapa de atualização dada pela Equação 2.20 não pode ser feita separadamente, em paralelo, para cada \mathbf{x}_i .

2.6 ADMM

O *Alternating Direction Method of Multipliers* (ADMM) combina as vantagens do DD com o MM. O método busca solução para problemas na seguinte forma

$$\begin{aligned} & \underset{\mathbf{x}, \mathbf{z}}{\text{minimizar}} && f(\mathbf{x}) + g(\mathbf{z}) \\ & \text{sujeito a} && \mathbf{Ax} + \mathbf{Bz} = \mathbf{c}, \end{aligned} \quad (2.22)$$

onde $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{z} \in \mathbb{R}^m$, $\mathbf{A} \in \mathbb{R}^{p \times n}$, $\mathbf{B} \in \mathbb{R}^{p \times m}$, $\mathbf{c} \in \mathbb{R}^p$, e f e g são funções convexas. Esse problema (Equação 2.22) é também chamado de ADMM de 2 blocos de variáveis. Note que a função objetivo é separável em dois conjuntos de variáveis. Como no MM, forma-se o Lagrangiano aumentado conforme

$$L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^\top (\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}) + \frac{\rho}{2} \|\mathbf{Ax} + \mathbf{Bz} - \mathbf{c}\|_2^2. \quad (2.23)$$

O algoritmo é muito similar ao DAM e MM. Na iteração k , busca-se a minimização em \mathbf{x} , depois em \mathbf{z} e finalmente atualiza-se a variável dual \mathbf{y} , deixando as outras variáveis constantes durante cada minimização. Como no MM, a variável dual é atualizada com um tamanho de passo igual ao parâmetro do Lagrangiano aumentado ρ . Logo,

$$\mathbf{x}^{k+1} := \arg \min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^k, \mathbf{y}^k) \quad (2.24)$$

$$\mathbf{z}^{k+1} := \arg \min_{\mathbf{z}} L_\rho(\mathbf{x}^{k+1}, \mathbf{z}, \mathbf{y}^k) \quad (2.25)$$

$$\mathbf{y}^{k+1} := \mathbf{y}^k + \rho (\mathbf{Ax}^{k+1} + \mathbf{Bz}^{k+1} - \mathbf{c}). \quad (2.26)$$

2.7 Global Variable Consensus ADMM

Considere o seguinte problema, com uma única variável global, e função objetivo dividida em N partes:

$$\underset{\mathbf{x}}{\text{minimizar}} \quad f(\mathbf{x}) = \sum_{i=1}^N f_i(\mathbf{x}), \quad (2.27)$$

onde $\mathbf{x} \in \mathbb{R}^n$, e $f_i : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ é convexa. O objetivo é resolver o problema dado pela Equação 2.27 de tal forma que cada termo possa ser tratado por seu próprio elemento de processamento, tal como uma *thread* ou processador.

Esse problema pode ser escrito com variáveis locais $\mathbf{x}_i \in \mathbb{R}^n$ e uma variável global comum \mathbf{z} (BOYD, 2010):

$$\begin{aligned} & \underset{\mathbf{x}_i}{\text{minimizar}} && \sum_{i=1}^N f_i(\mathbf{x}_i) \\ & \text{sujeito a} && \mathbf{x}_i - \mathbf{z} = 0, \quad i = 1, \dots, N. \end{aligned} \quad (2.28)$$

Tal problema é chamado de *global consensus*, já que a restrição é que todas as variáveis locais devem concordar, isto é, serem iguais.

O Lagrangiano aumentado da Equação 2.28 é dado por

$$L_\rho(\{\mathbf{x}_i\}, \mathbf{z}, \{\mathbf{y}_i\}) = \sum_{i=1}^N \left(f_i(\mathbf{x}_i) + \mathbf{y}_i^\top (\mathbf{x}_i - \mathbf{z}) + (\rho/2) \|\mathbf{x}_i - \mathbf{z}\|_2^2 \right). \quad (2.29)$$

O algoritmo *global variable consensus* ADMM (ou simplesmente *consensus* ADMM) consiste nas seguintes atualizações

$$\mathbf{x}_i^{k+1} := \arg \min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + (\mathbf{y}_i^k)^\top (\mathbf{x}_i - \mathbf{z}^k) + \frac{\rho}{2} \|\mathbf{x}_i - \mathbf{z}^k\|_2^2 \quad (2.30)$$

$$\mathbf{z}^{k+1} := \frac{1}{N} \sum_{i=1}^N \left(\mathbf{x}_i^{k+1} + \frac{1}{\rho} \mathbf{y}_i^k \right) \quad (2.31)$$

$$\mathbf{y}_i^{k+1} := \mathbf{y}_i^k + \rho (\mathbf{x}_i^{k+1} - \mathbf{z}^{k+1}). \quad (2.32)$$

Note que o primeiro e último passo são feitos de forma independente para cada $i = 1, \dots, N$.

2.8 Extreme Learning Machine

Huang, Zhu e Siew (2006) propuseram *Extreme Learning Machine* (ELM) para SLFN (*Single Hidden Layer Feedforward Neural Networks*), onde apenas os pesos de saída precisam ser determinados. Nesse caso, os parâmetros dos nós ocultos são escolhidos de forma aleatória e os pesos de saída são obtidos usando a inversa generalizada de Moore-Penrose.

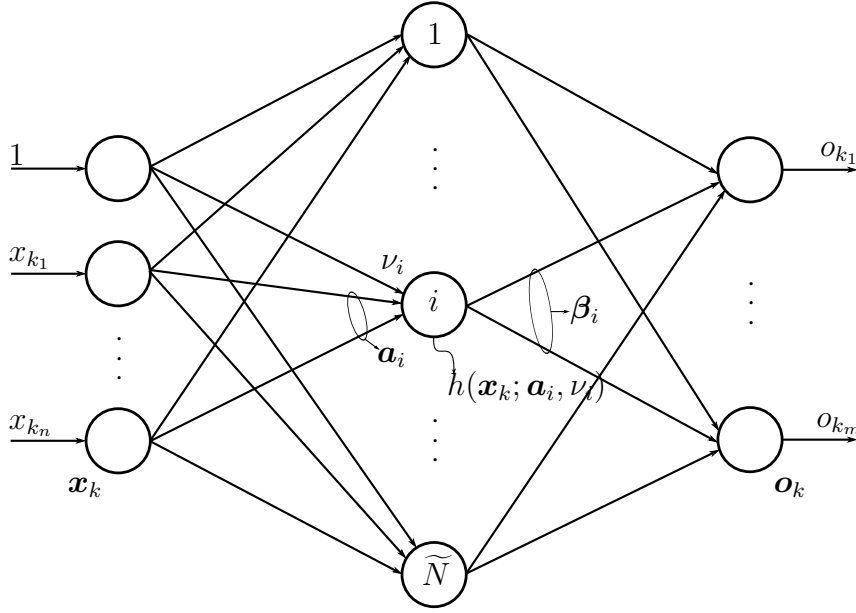
Para N amostras $\{(\mathbf{x}_k, \mathbf{t}_k)\}_{k=1}^N$, onde $\mathbf{x}_k \in \mathbb{R}^n$ é a k -ésima entrada e $\mathbf{t}_k \in \mathbb{R}^m$ a saída correspondente, a SLFN com \widetilde{N} nós ocultos e função de ativação $h(\cdot)$ é descrita como

$$\mathbf{o}_k = \sum_{i=1}^{\widetilde{N}} \beta_i h(\mathbf{x}_k; \mathbf{a}_i, \nu_i), \quad (2.33)$$

onde $\mathbf{a}_i \in \mathbb{R}^n$ e $\nu_i \in \mathbb{R}$ são, no contexto de ELM, os parâmetros que são aleatoriamente atribuídos do i -ésimo nó oculto, e $\beta_i \in \mathbb{R}^m$ é o vetor de pesos que conecta o i -ésimo nó oculto ao nó de saída.

A Figura 2 exibe uma arquitetura típica para ser treinada pelo ELM em problemas com m -saídas, n entradas, e \widetilde{N} nós aleatórios.

Figura 2 – Ilustração de uma arquitetura SFLN com n entradas, \widetilde{N} nós ocultos, e m saídas.



Fonte: Produção do próprio autor.

Seja $\mathbf{B} = (\beta_1, \beta_2, \dots, \beta_{\widetilde{N}})^\top$, $\mathbf{T} = (t_1, t_2, \dots, t_N)^\top$, e

$$\mathbf{H}(\mathbf{X}; \mathbf{A}, \boldsymbol{\nu}) = \begin{bmatrix} h(\mathbf{x}_1; \mathbf{a}_1, \nu_1) & \cdots & h(\mathbf{x}_1; \mathbf{a}_{\widetilde{N}}, \nu_{\widetilde{N}}) \\ \vdots & \ddots & \vdots \\ h(\mathbf{x}_N; \mathbf{a}_1, \nu_1) & \cdots & h(\mathbf{x}_N; \mathbf{a}_{\widetilde{N}}, \nu_{\widetilde{N}}) \end{bmatrix}, \quad (2.34)$$

onde $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N)^\top$, $\mathbf{A} = (\mathbf{a}_1, \dots, \mathbf{a}_{\widetilde{N}})^\top$ e $\boldsymbol{\nu} = (\nu_1, \dots, \nu_{\widetilde{N}})^\top$.

Se uma SLFN com \widetilde{N} nós ocultos pode aproximar N amostras dadas, $\{(\mathbf{x}_k, \mathbf{t}_k)\}_{k=1}^N$, com erro zero, i.e. $\mathbf{o}_k = \mathbf{t}_k$, então a Equação 2.33 pode ser expressa compactamente por

$$\mathbf{H}\mathbf{B} = \mathbf{T}. \quad (2.35)$$

Os pesos de saída podem ser obtidos pela solução dos mínimos quadrados de norma mínima, $\mathbf{B} = \mathbf{H}^\dagger \mathbf{T}$, onde \mathbf{H}^\dagger é a inversa generalizada de Moore-Penrose (RAO; MITRA, 1971) da matriz \mathbf{H} . A inversa generalizada de Moore-Penrose pode ser determinada usando o método de projeção ortogonal (RAO; MITRA, 1971) em dois casos: quando $\mathbf{H}^\top \mathbf{H}$ é não singular e $\mathbf{H}^\dagger = (\mathbf{H}^\top \mathbf{H})^{-1} \mathbf{H}^\top$, ou quando $\mathbf{H}\mathbf{H}^\top$ é não singular e $\mathbf{H}^\dagger = \mathbf{H}^\top (\mathbf{H}\mathbf{H}^\top)^{-1}$.

2.9 Regularized Extreme Learning Machine

A fim de se obter uma solução mais estável e uma melhor generalização, reduzindo o *overfitting*, Deng, Zheng e Chen (2009) propuseram o *Weighted Regularized Extreme*

Learning Machine (WR-ELM), que consiste em adicionar um valor positivo à diagonal de $\mathbf{H}^\top \mathbf{H}$ ou $\mathbf{H} \mathbf{H}^\top$ e ponderar o vetor de erros. O seguinte problema de otimização foi proposto por Deng, Zheng e Chen (2009):

$$\begin{aligned} & \underset{\boldsymbol{\beta}}{\text{minimizar}} \quad \frac{1}{2} \|\boldsymbol{\beta}\|_2^2 + \frac{C}{2} \|\mathbf{D}\boldsymbol{\epsilon}\|_2^2 \\ & \text{sujeito a} \quad \mathbf{H}\boldsymbol{\beta} - \mathbf{y} = \boldsymbol{\epsilon}, \end{aligned} \quad (2.36)$$

onde $\boldsymbol{\epsilon}$ é o vetor de erro, C é o parâmetro de regularização, $\boldsymbol{\beta} = (\beta_1, \beta_2, \dots, \beta_{\tilde{N}})^\top$ é o vetor que conecta os \tilde{N} nós ocultos à saída $\mathbf{y} = (y_1, y_2, \dots, y_N)^\top$, e \mathbf{D} é uma matriz diagonal de pesos adicionada ao modelo para melhorar a robustez em relação aos *outliers*. A solução de $\boldsymbol{\beta}$ é dada por

$$\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}^\top \mathbf{D}^2 \mathbf{H} \right)^{-1} \mathbf{H}^\top \mathbf{D}^2 \mathbf{y}, \quad (2.37)$$

quando $\tilde{N} \ll N$. Quando \mathbf{D} é igual a matriz identidade, $\boldsymbol{\beta}$ é determinado por

$$\boldsymbol{\beta} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}^\top \mathbf{H} \right)^{-1} \mathbf{H}^\top \mathbf{y}, \quad (2.38)$$

que é chamado de *Unweighted Regularized Extreme Learning Machine* (UWR-ELM). A avaliação do desempenho apresentada em Deng, Zheng e Chen (2009) faz uso de SLFN com função de ativação sigmoideal e nós aditivos. Huang et al. (2012) estenderam o trabalho feito em (DENG; ZHENG; CHEN, 2009) para generalizar a SLFN com diferentes tipos de funções de ativação na camada oculta, bem como a utilização de *kernels*. Além disso, Huang et al. (2012) apresentaram uma solução alternativa para o caso onde $\tilde{N} > N$.

Martínez-Martínez et al. (2011) consideraram algumas regularizações para regressão de mínimos quadrados na determinação dos pesos de saída. Em particular, *lasso* (TIBSHIRANI, 1996), regularização *ridge* (HOERL; KENNARD, 1970), e o *elastic net* (ZOU; HASTIE, 2005) são abordados. Os três métodos de regularização são descritos e generalizados na seguinte forma

$$\underset{(\beta_0, \boldsymbol{\beta}) \in \mathbb{R}^{\tilde{N}+1}}{\text{minimizar}} \quad \frac{1}{2N} \sum_{k=1}^N \left(y_k - \beta_0 - \mathbf{h}_k^\top \boldsymbol{\beta} \right)^2 + \lambda P_\alpha(\boldsymbol{\beta}), \quad (2.39)$$

onde \mathbf{h}_k^\top é a k -ésima linha da matriz \mathbf{H} , λ é o parâmetro de regularização, e

$$P_\alpha(\boldsymbol{\beta}) = \sum_{i=1}^{\tilde{N}} \left[\frac{1}{2} (1 - \alpha) \beta_i^2 + \alpha |\beta_i| \right] \quad (2.40)$$

é a regularização *elastic net*. P_α é o compromisso entre a regularização *ridge* ($\alpha = 0$) e *lasso* ($\alpha = 1$) (MARTÍNEZ-MARTÍNEZ et al., 2011; FRIEDMAN; HASTIE; TIBSHIRANI, 2010). A ideia de se utilizar a regularização *elastic net* é de identificar o grau de relevância dos pesos que conectam os nós da camada oculta à saída. Assim, uma rede mais compacta

pode ser obtida através da remoção de nós irrelevantes da camada oculta sem comprometer a habilidade de generalização da rede (MARTÍNEZ-MARTÍNEZ et al., 2011).

Quando $\beta_0 = 0$, o estimador para o caso de regularização *ridge* ($\alpha = 0$) é

$$\beta_{\text{ridge}} = (\lambda \mathbf{I} + \mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{y}, \quad (2.41)$$

que é a mesma solução da Equação 2.38, a *unweighted regularized* ELM. Conforme apontado por Martínez-Martínez et al. (2011), a regularização *ridge* não possui a habilidade de podar a arquitetura da rede de forma eficiente, embora apresente boa generalização para problemas de classificação.

2.10 Outlier Robust Extreme Learning Machine

Em busca por uma abordagem que fosse robusta à *outliers* Zhang e Luo (2015) propuseram o *Outlier Robust Extreme Learning Machine* (OR-ELM).

Em geral, *outliers* ocupam uma fração pequena de todo conjunto de amostras de treino. Para o erro de treino, ϵ , essa característica pode ser descrita como esparsidade. Já é bem estabelecido na literatura que esparsidade pode ser melhor contemplada pelo uso da norma ℓ_0 ao invés da ℓ_2 . Portanto, a proposta dos autores é forçar a esparsidade no erro através da norma ℓ_0 , i.e.,

$$\begin{aligned} \underset{\beta}{\text{minimizar}} \quad & \|\epsilon\|_0 + \frac{1}{\tau} \|\beta\|_2^2 \\ \text{sujeito a} \quad & \mathbf{H}\beta - \mathbf{y} = \epsilon. \end{aligned} \quad (2.42)$$

Entretanto, a Equação 2.42 é um problema de otimização não convexo e NP-completo. Uma maneira de contornar esse problema é realizar um relaxamento convexo sem perder a característica de esparsidade. Isso pode ser alcançado substituindo a norma ℓ_0 pela norma ℓ_1 . Como resultado, a formulação do OR-ELM é dada por

$$\begin{aligned} \underset{\beta}{\text{minimizar}} \quad & \|\epsilon\|_1 + \frac{1}{\tau} \|\beta\|_2^2 \\ \text{sujeito a} \quad & \mathbf{H}\beta - \mathbf{y} = \epsilon. \end{aligned} \quad (2.43)$$

Para resolver o problema da Equação 2.43, Zhang e Luo (2015) propuseram o uso do método *augmented Lagrange multiplier* (ALM).

Embora o WR-ELM e OR-ELM apresentem bons resultados para problemas com *outliers*, os pesos de saída são densos², e um número apropriado de neurônios precisa ser escolhido a priori. Além disso, conforme comentado na seção 2.5, o ALM não pode ser implementado de forma paralela.

² Todos os elementos de β são não nulos.

3 Método Proposto

Os métodos regularizados para ELM, em geral, se limitam a tratar problemas que possuem uma única saída. Neste capítulo uma generalização para contemplar problemas de múltiplas saídas é feita. Com essa generalização, dois novos métodos são definidos, o GR-ELM e GOR-ELM. O primeiro é uma generalização do R-ELM e será aplicado para problemas de classificação multiclasse. Já o GOR-ELM é uma generalização do OR-ELM e será aplicado em problemas de regressão com múltiplas saídas contaminadas com *outliers*.

De forma mais geral, a generalização do ELM regularizado pode ser escrita como o seguinte problema de otimização

$$\underset{\mathbf{B}}{\text{minimizar}} \quad \gamma_1 \mathcal{L}(\mathbf{H}\mathbf{B}, \mathbf{T}) + \gamma_2 \mathcal{P}(\mathbf{B}), \quad (3.1)$$

onde $\mathcal{L}(\cdot)$ é uma função de perda, $\mathcal{P}(\cdot)$ é uma função de penalização dos pesos de saída, γ_1 e γ_2 são parâmetros de regularização. No contexto de aplicações em problemas de múltiplas saídas, a função perda e função de penalização podem ser normas matriciais. De fato, nesta Tese, a função perda será uma norma matricial e a função de penalização dos pesos é uma função composta por uma ou mais normas matriciais. Os parâmetros de regularização servem para balancear a importância dada por cada uma das parcelas do problema de otimização. Em geral, esses parâmetros são obtidos através de validação cruzada, ou pela experiência do usuário.

Note que dependendo da aplicação, normas diferentes podem ser utilizadas para melhor tratar o problema. Para problemas onde há *outliers* nas saídas, a norma $\ell_{2,1}$ pode ser usada como função de perda. Na ausência de *outliers*, mas com presença de ruídos, a norma Frobenius pode ser utilizada como função de perda.

Para a função de penalização dos pesos, pode-se utilizar a norma Frobenius para melhorar a generalização. Se o objetivo é também buscar uma rede mais compacta, além da norma Frobenius, pode-se adicionar a norma $\ell_{2,1}$ na função de penalização dos pesos.

Nos exemplos citados até o momento neste capítulo, as soluções dos problemas de otimização dadas pela Equação 3.1 usando as normas matriciais citadas, podem ser obtidas utilizando o ADMM. Uma vantagem de se utilizar o ADMM é que esse pode ser usado em problemas distribuídos com poucas alterações.

O resto do capítulo está organizado da seguinte forma: primeiro, generaliza-se o R-ELM para problemas de classificação multiclasse. Apresenta-se as atualizações para o ADMM que foram utilizadas para resolver o problema de otimização resultante do método proposto, GR-ELM. Seguidamente, apresenta-se uma alternativa para solucionar o GR-ELM em tarefas de aprendizado com volume de dados grande usando *consensus*

ADMM. Posteriormente, uma generalização do OR-ELM é feita para tratar problemas de regressão de múltiplas saídas que possuem *outliers* e as respectivas atualizações do ADMM são apresentadas. Por fim é definido o critério de parada e as generalizações alcançadas.

3.1 Generalized Regularized Extreme Learning Machine

Ao invés de aplicar o R-ELM para cada coluna de \mathbf{T} , aqui, busca-se por uma solução conjuntamente esparsa. A ideia principal da abordagem proposta nesta seção é determinar um conjunto de soluções que compartilham um suporte comum, não nulo, tal que uma rede mais compacta seja obtida (INABA et al., 2017). O seguinte problema de otimização é proposto

$$\underset{\mathbf{B}}{\text{minimizar}} \quad \frac{C}{2} \|\mathbf{HB} - \mathbf{T}\|_F^2 + \lambda_1 \|\mathbf{B}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2, \quad (3.2)$$

onde $\|\cdot\|_F$ é a norma Frobenius, $\|\mathbf{B}\|_{2,1} = \sum_{i=1}^{\tilde{N}} \|\mathbf{b}_{i,\cdot}\|$ e $\mathbf{b}_{i,\cdot}$ é a i -ésima linha de \mathbf{B} . Neste trabalho utiliza-se o *Alternating Direction Method of Multipliers* (ADMM) (BOYD, 2010) para resolver a Equação 3.2.

Como o foco é tratar problemas de classificação multiclasse, a norma Frobenius é a extensão natural da norma ℓ_2 para lidar com múltiplas saídas. A norma $\ell_{2,1}$ em \mathbf{B} é utilizada pois deseja-se uma rede mais compacta. Compacta no sentido de que se deseja a menor quantidade de neurônios na camada oculta sem comprometer a acurácia de treino e teste. Se fosse imposto a norma ℓ_1 em cada coluna de \mathbf{B} , suas colunas seriam esparsas, mas não necessariamente todos os elementos de uma linha seriam iguais a zero. Quando uma linha de \mathbf{B} possui todos os elementos iguais a zero, pode-se eliminar o neurônio associado a essa linha. A regularização usando a norma $\ell_{2,1}$ possui o papel de eliminar os neurônios ao fazer todos os elementos de uma mesma linha iguais a zero, como ilustrado na Figura 3.

O problema de minimização estabelecido na Equação 3.2 é equivalente ao seguinte problema restrito

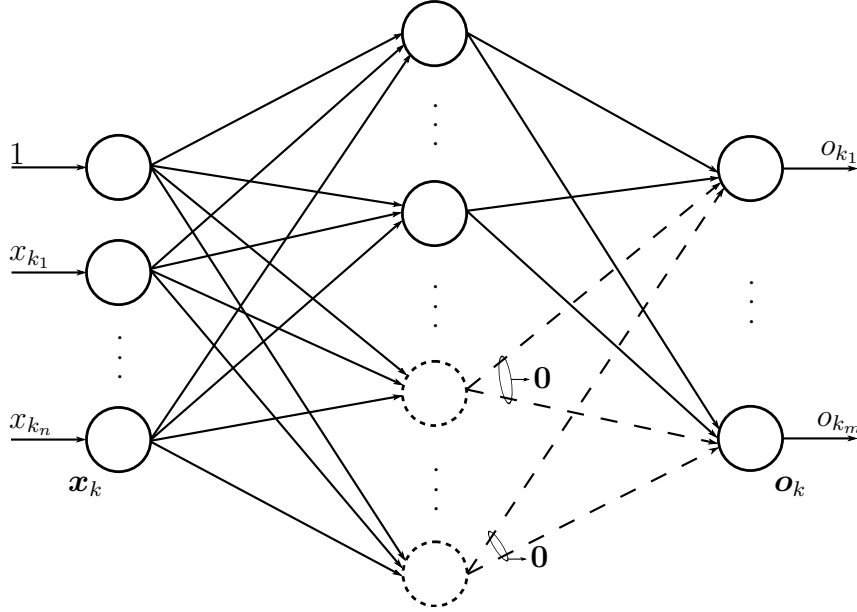
$$\begin{aligned} \underset{\mathbf{B}, \mathbf{Z}}{\text{minimizar}} \quad & \frac{C}{2} \|\mathbf{HB} - \mathbf{T}\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} \\ \text{sujeito a} \quad & \mathbf{B} - \mathbf{Z} = \mathbf{0}, \end{aligned} \quad (3.3)$$

onde a função objetivo é separável em \mathbf{B} e \mathbf{Z} . Em cada iteração do ADMM, uma minimização alternada do Lagrangiano aumentado sobre \mathbf{B} e \mathbf{Z} é efetuada.

A função Lagrangiano aumentado da Equação 3.3 é

$$\begin{aligned} L(\mathbf{B}, \mathbf{Z}, \mathbf{Y}) = & \frac{C}{2} \|\mathbf{HB} - \mathbf{T}\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 \\ & + \lambda_1 \|\mathbf{Z}\|_{2,1} + \frac{\rho}{2} \|\mathbf{B} - \mathbf{Z}\|_F^2 + \langle \mathbf{Y}, \mathbf{B} - \mathbf{Z} \rangle, \end{aligned} \quad (3.4)$$

Figura 3 – Eliminação de neurônios que possuem todos os pesos associados ao mesmo iguais a zero, pela regularização com a norma $\ell_{2,1}$ em uma arquitetura típica para GR-ELM com n entradas e m saídas.



Fonte: Produção do próprio autor.

onde \mathbf{Y} é o multiplicador de Lagrange, e $\rho > 0$ é o parâmetro de regularização.

Usando a variável dual escalada, é possível escrever o Lagrangiano aumentado em uma forma diferente, como se segue (ver Apêndice C),

$$L(\mathbf{B}, \mathbf{Z}, \mathbf{U}) = \frac{C}{2} \|\mathbf{H}\mathbf{B} - \mathbf{T}\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} + \frac{\rho}{2} \|\mathbf{B} - \mathbf{Z} + \mathbf{U}\|_F^2 - \frac{\rho}{2} \|\mathbf{U}\|_F^2, \quad (3.5)$$

onde $\mathbf{U} = (1/\rho)\mathbf{Y}$ é a variável dual escalada.

Na iteração k , o ADMM consiste nas seguintes regras de atualização para

1. \mathbf{B}^{k+1} , tem-se o seguinte subproblema

$$\mathbf{B}^{k+1} := \arg \min_{\mathbf{B}} \frac{C}{2} \|\mathbf{H}\mathbf{B} - \mathbf{T}\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \frac{\rho}{2} \|\mathbf{B} - \mathbf{Z}^k + \mathbf{U}^k\|_F^2. \quad (3.6)$$

Fazendo o gradiente da função objetivo com respeito a \mathbf{B} igual a $\mathbf{0}$, tem-se (ver Apêndice C)

$$\mathbf{H}^\top (\mathbf{H}\mathbf{B} - \mathbf{T}) + \frac{\lambda_2}{C} \mathbf{B} + \frac{\rho}{C} (\mathbf{B} - \mathbf{Z}^k + \mathbf{U}^k) = \mathbf{0}, \quad (3.7)$$

e a solução para Equação 3.6 é

$$\mathbf{B} = (\mathbf{H}^\top \mathbf{H} + \eta \mathbf{I})^{-1} \left[\mathbf{H}^\top \mathbf{T} + \frac{\rho}{C} (\mathbf{Z}^k - \mathbf{U}^k) \right], \quad (3.8)$$

onde $\eta = (\lambda_2 + \rho) / C$.

2. \mathbf{Z}^{k+1} , tem-se o seguinte subproblema

$$\mathbf{Z}^{k+1} := \arg \min_{\mathbf{Z}} \frac{\lambda_1}{\rho} \|\mathbf{Z}\|_{2,1} + \frac{1}{2} \|\mathbf{B}^{k+1} - \mathbf{Z} + \mathbf{U}^k\|_F^2. \quad (3.9)$$

Esse problema de otimização é equivalente aos seguintes problemas de otimização por linha e para todas as linhas de \mathbf{Z} , \mathbf{B} e \mathbf{U} :

$$\mathbf{z}_{i,\cdot}^{k+1} := \arg \min_{\mathbf{z}_{i,\cdot}} \frac{\lambda_1}{\rho} \|\mathbf{z}_{i,\cdot}\|_2 + \frac{1}{2} \|\mathbf{b}_{i,\cdot}^{k+1} - \mathbf{z}_{i,\cdot} + \mathbf{u}_{i,\cdot}^k\|_2^2. \quad (3.10)$$

A Equação 3.10 possui solução fechada (BOYD, 2010; DONOHO; JOHNSTONE; MONTANARI, 2013) dada por

$$\mathbf{z}_{i,\cdot}^{k+1} = S_{\frac{\lambda_1}{\rho}} \left(\mathbf{b}_{i,\cdot}^{k+1} + \mathbf{u}_{i,\cdot}^k \right), \quad (3.11)$$

onde $\kappa \in \mathbb{R}$ e $S_\kappa : \mathbb{R}^m \rightarrow \mathbb{R}^m$ é o operador *block soft-thresholding* (BOYD, 2010; DONOHO; JOHNSTONE; MONTANARI, 2013) definido como

$$S_\kappa(\mathbf{a}) = \left(1 - \frac{\kappa}{\|\mathbf{a}\|_2} \right)_+ \mathbf{a}, \quad (3.12)$$

com $S_\kappa(\mathbf{0}) = \mathbf{0}$ e $(d)_+ \equiv \max(0, d)$. Note que $S_\kappa(\mathbf{a})$ encolhe (*shrink*) o seu argumento \mathbf{a} para $\mathbf{0}$ se $\|\mathbf{a}\|_2 \leq \kappa$ e o move por uma quantidade κ em direção à origem caso contrário. O operador *soft-thresholding* pode ser visto como um caso particular do operador *block soft-thresholding* quando \mathbf{a} é um escalar. Para mais detalhes do operador *soft-thresholding* ver Apêndice B.

Seja $\mathcal{S}_\kappa(\mathbf{A})$ o operador que aplica o *block soft-thresholding*, Equação 3.12, em cada linha de \mathbf{A} . É possível escrever a solução para Equação 3.9 como

$$\mathbf{Z}^{k+1} = \mathcal{S}_{\frac{\lambda_1}{\rho}} \left(\mathbf{B}^{k+1} + \mathbf{U}^k \right). \quad (3.13)$$

3. \mathbf{U}^{k+1} , o multiplicador de Lagrange é atualizado por

$$\mathbf{U}^{k+1} := \mathbf{U}^k + \mathbf{B}^{k+1} - \mathbf{Z}^{k+1}. \quad (3.14)$$

O algoritmo 1 resume o GR-ELM.

Algoritmo 1: GR-ELM: Generalized Regularized Extreme Learning Machine

- 1 **Requer:** Amostras de treino $(\mathbf{x}_p, \mathbf{t}_p)$, com $p = 1, \dots, N$, função de ativação $h(\cdot)$ (no caso, sigmoidal), e parâmetros de regularização λ_1 , λ_2 , C , e ρ .
 - 2 **Inicializar:** \mathbf{B}^0 , \mathbf{U}^0 , \mathbf{Z}^0 , e $k = 0$.
 - 3 Atribuir os parâmetros \mathbf{A} e ν aleatoriamente.
 - 4 Calcular a matriz de saída da camada oculta \mathbf{H} usando Equação 2.34.
 - 5 **repita**
 - 6 $\mathbf{B}^{k+1} := \arg \min_{\mathbf{B}} \frac{C}{2} \|\mathbf{H}\mathbf{B} - \mathbf{T}\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \frac{\rho}{2} \|\mathbf{B} - \mathbf{Z}^k + \mathbf{U}^k\|_F^2$
 - 7 $\mathbf{Z}^{k+1} := \arg \min_{\mathbf{Z}} \frac{\lambda_1}{\rho} \|\mathbf{Z}\|_{2,1} + \frac{1}{2} \|\mathbf{B}^{k+1} - \mathbf{Z} + \mathbf{U}^k\|_F^2$
 - 8 $\mathbf{U}^{k+1} := \mathbf{U}^k + \mathbf{B}^{k+1} - \mathbf{Z}^{k+1}$
 - 9 $k = k + 1$
 - 10 **até atender o critério de parada.**
-

As derivações matemáticas para determinação das atualizações do ADMM encontram-se no Apêndice C.

3.2 Distributed GR-ELM

Os problemas tratados até o momento pelo GR-ELM se limitam à aplicações em que os dados estão armazenados em apenas um computador. No entanto, existem aplicações onde os dados são armazenados ou coletados de forma distribuída, ou existem tantos dados que processá-los em um único computador é impossível ou inconveniente. Nesta seção discute-se uma abordagem para resolver o GR-ELM (Equação 3.2) quando os dados são armazenados de forma distribuída e/ou quando o número de amostras de treino é grande, com um número modesto de neurônios. Utiliza-se o *global variable consensus* ADMM (BOYD, 2010) para resolver o problema de forma distribuída, de modo que cada processador é responsável pelo processamento de um subconjunto dos dados de treino.

Particiona-se o conjunto de treino em M partes. Seja $\mathbf{X} = [\mathbf{X}_1^\top, \dots, \mathbf{X}_M^\top]^\top$ e $\mathbf{T} = [\mathbf{T}_1^\top, \dots, \mathbf{T}_M^\top]^\top$, onde $\mathbf{X}_j \in \mathbb{R}^{N_j \times n}$, $\mathbf{T}_j \in \mathbb{R}^{N_j \times m}$, e $\sum_{j=1}^M N_j = N$. Considera-se que cada processador j possui acesso aos parâmetros dos nós da camada oculta, de forma que possam gerar a matriz $\mathbf{H}_j \in \mathbb{R}^{N_j \times \tilde{N}}$ pela Equação 2.34. Logo, \mathbf{H}_j e \mathbf{T}_j será processado pelo j -ésimo processador.

Pode-se escrever a Equação 3.3 na forma *consensus*

$$\begin{aligned}
 & \underset{\mathbf{B}, \mathbf{Z}}{\text{minimizar}} && \frac{C}{2} \sum_{j=1}^M \|\mathbf{H}_j \mathbf{B}_j - \mathbf{T}_j\|_F^2 + \frac{\lambda_2}{2} \sum_{j=1}^M \|\mathbf{B}_j\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} \\
 & \text{sujeito a} && \mathbf{B}_j - \mathbf{Z} = \mathbf{0}, \quad j = 1, \dots, M,
 \end{aligned} \tag{3.15}$$

onde as variáveis locais $\mathbf{B}_j \in \mathbb{R}^{\tilde{N} \times m}$ e a variável global $\mathbf{Z} \in \mathbb{R}^{\tilde{N} \times m}$. O algoritmo ADMM resultante, na forma escalada, é

$$\mathbf{B}_j^{k+1} := \arg \min_{\mathbf{B}_j} \frac{C}{2} \|\mathbf{H}_j \mathbf{B}_j - \mathbf{T}_j\|_F^2 + \frac{\lambda}{2} \|\mathbf{B}_j\|_F^2 + \frac{\rho}{2} \|\mathbf{B}_j - \mathbf{Z}^k + \mathbf{U}_j^k\|_F^2 \quad (3.16)$$

$$\mathbf{Z}^{k+1} := \arg \min_{\mathbf{Z}} \frac{\lambda_1}{\rho} \|\mathbf{Z}\|_{2,1} + \frac{1}{2} \sum_{j=1}^M \|\mathbf{B}_j^{k+1} - \mathbf{Z} + \mathbf{U}_j^k\|_F^2 \quad (3.17)$$

$$\mathbf{U}_j^{k+1} := \mathbf{U}_j^k + \mathbf{B}_j^{k+1} - \mathbf{Z}^{k+1}. \quad (3.18)$$

Note que é possível expressar a atualização de \mathbf{Z} (Equação 3.17) como (BOYD, 2010)

$$\mathbf{Z}^{k+1} := \arg \min_{\mathbf{Z}} \left(\frac{\lambda_1}{M\rho} \|\mathbf{Z}\|_{2,1} + \frac{1}{2} \|\bar{\mathbf{B}}^{k+1} - \mathbf{Z} + \bar{\mathbf{U}}^k\|_F^2 \right), \quad (3.19)$$

onde $\bar{\mathbf{B}}^{k+1} = 1/M \sum_{j=1}^M \mathbf{B}_j^{k+1}$ e $\bar{\mathbf{U}}^{k+1} = 1/M \sum_{j=1}^M \mathbf{U}_j^{k+1}$. As soluções para Equação 3.16 e Equação 3.19 são

$$\mathbf{B}_j = \left(\mathbf{H}_j^\top \mathbf{H}_j + \frac{\lambda_2 + \rho}{C} \mathbf{I} \right)^{-1} \left[\mathbf{H}_j^\top \mathbf{T}_j + \frac{\rho}{C} (\mathbf{Z}^k - \mathbf{U}_j^k) \right] \quad (3.20)$$

e

$$\mathbf{Z}^{k+1} = \mathcal{S}_{\frac{\lambda_1}{M\rho}} \left(\bar{\mathbf{B}}^{k+1} + \bar{\mathbf{U}}^k \right), \quad (3.21)$$

respectivamente, onde \mathcal{S}_κ é o operador *block soft-thresholding* para matrizes conforme definida anteriormente. Implementou-se o *Distributed GR-ELM* (DGR-ELM) com *Message Passing Interface* (MPI) usando *Single Program, Multiple Data* (SPMD). O algoritmo 2 resume o DGR-ELM.

Algoritmo 2: DGR-ELM: Distributed Generalized Regularized Extreme Learning Machine

- 1 **Requer:** M processadores, com cada processador j armazenando amostras de treino $(\mathbf{X}_j, \mathbf{T}_j) = \{\mathbf{x}_{pj}, \mathbf{t}_{pj}\}_{p_j=1}^{N_j}$, parâmetros de regularização λ_1 , λ_2 , C , e ρ , parâmetros \mathbf{A} e $\boldsymbol{\nu}$, e função de ativação $h(\cdot)$.
 - 2 **Inicializar:** M processadores, juntamente com \mathbf{B}_j , \mathbf{U}_j e \mathbf{Z} .
 - 3 Calcular a j -ésima matriz de saída da camada oculta $\mathbf{H}_j(\mathbf{X}_j, \mathbf{A}, \boldsymbol{\nu})$ usando Equação 2.34.
 - 4 **repita**
 - 5 $\mathbf{U}_j := \mathbf{U}_j + \mathbf{B}_j - \mathbf{Z}$
 - 6 $\mathbf{B}_j := \arg \min_{\mathbf{B}_j} \frac{C}{2} \|\mathbf{H}_j \mathbf{B}_j - \mathbf{T}_j\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}_j\|_F^2 + \frac{\rho}{2} \|\mathbf{B}_j - \mathbf{Z} + \mathbf{U}_j\|_F^2$
 - 7 $\mathbf{W} := \mathbf{B}_j + \mathbf{U}_j$
 - 8 *Allreduce* \mathbf{W}
 - 9 $\mathbf{Z} := \arg \min_{\mathbf{Z}} \frac{\lambda_1}{M\rho} \|\mathbf{Z}\|_{2,1} + \frac{1}{2} \|\mathbf{W} - \mathbf{Z}\|_F^2$
 - 10 **até** atender o critério de parada.
-

No passo 1, assume-se que cada processador possui acesso aos parâmetros de regularização, função de ativação e parâmetros \mathbf{A} e $\boldsymbol{\nu}$. Entretanto, os parâmetros e função de ativação podem ser gerados/armazenados em apenas um processador e transmitido para os demais processadores.

No passo 8, *Allreduce* denota a operação `MPI_Allreduce` para calcular a soma global sobre todos os processadores e armazenar o resultado $\mathbf{W} = \sum_{j=1}^M (\mathbf{B}_j + \mathbf{U}_j) = M (\bar{\mathbf{B}} + \bar{\mathbf{U}})$ em todos os processadores. No passo 9, todos os processadores (de forma redundante) calculam a atualização de \mathbf{Z} . Embora um único processador possa realizar essa operação e transmitir o resultado para os demais processadores, essa abordagem complica o código e em geral não é mais rápida (BOYD, 2010).

3.3 Generalized Outlier Robust Extreme Learning Machine

Regressão com múltiplas saídas, também conhecida como *multi-target regression* (MTR) refere-se aos problemas de predição de múltiplas saídas, de valores reais e de forma simultânea, a partir de um conjunto comum de entradas. Esses problemas ocorrem em diversas áreas do conhecimento incluindo modelagem ecológica, economia, energia, *data mining*, e análise de imagens médicas (GHOSH; BENGIO, 1997; DŽEROSKI; DEMŠAR; GRBOVIĆ, 2000; KOCEV et al., 2009; WANG et al., 2015; ZHEN et al., 2016; SPYROMITROS-XIOUFIS et al., 2016).

Uma abordagem natural para tratar esse tipo de problema é obter um modelo para cada saída, separadamente. Entretanto, um dos grandes desafios de MTR está na modelagem das correlações entre saídas e nas relações não-lineares de entrada-saída (ZHANG; YEUNG, 2010; ZHEN et al., 2017).

Com o advento da era do *big data*, problemas de MTR têm se tornado cada vez mais comuns no imenso volume de dados que são acumulados. Entretanto, inerente a esse grande volume de dados, outros problemas surgem. É cada vez mais comum dados com *outliers* devido à diferentes causas (HODGE; AUSTIN, 2004; ZHANG; LUO, 2015). Embora avanços tenham sido feitos no que tange a solução de problemas MTR, considerando a complexidade de relações entre saídas e entradas-saídas, pouco se tem feito em relação ao tratamento de problemas MTR com *outliers*.

Nesta seção, apresenta-se uma abordagem para tratar problemas de *Multi-Target Regression* (MTR) que possuem *outliers*. O seguinte problema de otimização é proposto

$$\underset{\mathbf{B}}{\text{minimizar}} \quad \tau \|\mathbf{HB} - \mathbf{T}\|_{2,1} + \lambda_1 \|\mathbf{B}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2. \quad (3.22)$$

Note que a única diferença entre o GR-ELM (Equação 3.2) e o GOR-ELM (Equação 3.22) é a substituição da norma Frobenius do erro $\mathbf{HB} - \mathbf{T}$ pela norma $\ell_{2,1}$.

A utilização da norma ℓ_1 no erro para tratar problemas de regressão é uma prática comum na literatura (XU et al., 2013; ZHANG; LUO, 2015; CHEN et al., 2017). A ideia do GOR-ELM é estender o OR-ELM para problemas de MTR. Contudo, ao invés de tratar cada saída de forma independente, o GOR-ELM considera a estrutura da saída na penalização. Problemas onde os *outliers* ocorrem nas saídas de forma estruturada³ são contemplados pelo GOR-ELM.

O problema de minimização do GOR-ELM (Equação 3.22) é equivalente ao seguinte problema restrito

$$\begin{aligned} \underset{\mathbf{E}, \mathbf{B}, \mathbf{Z}}{\text{minimizar}} \quad & \tau \|\mathbf{E}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} \\ \text{sujeito a} \quad & \mathbf{E} = \mathbf{H}\mathbf{B} - \mathbf{T} \\ & \mathbf{B} - \mathbf{Z} = \mathbf{0}, \end{aligned} \quad (3.23)$$

onde a função objetivo é separável em \mathbf{E} , \mathbf{B} e \mathbf{Z} . Como nos problemas anteriores, para solução do GOR-ELM utiliza-se o ADMM para resolver o problema de otimização. Em cada iteração do ADMM, uma minimização alternada do Lagrangiano aumentado sobre \mathbf{E} , \mathbf{B} e \mathbf{Z} é efetuada.

Note que é possível reescrever o problema Equação 3.23 como

$$\begin{aligned} \underset{\mathbf{E}, \mathbf{B}, \mathbf{Z}}{\text{minimizar}} \quad & \tau \|\mathbf{E}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} \\ \text{sujeito a} \quad & \tilde{\mathbf{A}}\mathbf{E} + \tilde{\mathbf{B}}\mathbf{B} + \tilde{\mathbf{C}}\mathbf{Z} + \tilde{\mathbf{D}} = \mathbf{0}, \end{aligned} \quad (3.24)$$

onde $\tilde{\mathbf{A}} = [-\mathbf{I}_N, \mathbf{0}_{N \times \tilde{N}}]^\top$, $\tilde{\mathbf{B}} = [\mathbf{H}^\top, \mathbf{I}_{\tilde{N}}]^\top$, $\tilde{\mathbf{C}} = [\mathbf{0}_{\tilde{N} \times N}, -\mathbf{I}_{\tilde{N}}]^\top$ e $\tilde{\mathbf{D}} = [-\mathbf{T}^\top, \mathbf{0}_{m \times \tilde{N}}]^\top$. Esse problema é também chamado de ADMM de 3 blocos de variáveis.

A função Lagrangiano aumentado da Equação 3.24 é

$$\begin{aligned} L(\mathbf{E}, \mathbf{B}, \mathbf{Z}, \mathbf{Y}) = & \tau \|\mathbf{E}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} \\ & + \frac{\rho}{2} \|\tilde{\mathbf{A}}\mathbf{E} + \tilde{\mathbf{B}}\mathbf{B} + \tilde{\mathbf{C}}\mathbf{Z} + \tilde{\mathbf{D}}\|_F^2 + \langle \mathbf{Y}, \tilde{\mathbf{A}}\mathbf{E} + \tilde{\mathbf{B}}\mathbf{B} + \tilde{\mathbf{C}}\mathbf{Z} + \tilde{\mathbf{D}} \rangle, \end{aligned} \quad (3.25)$$

onde \mathbf{Y} é o multiplicador de Lagrange, e $\rho > 0$ é o parâmetro de regularização.

Usando a variável dual escalada, o Lagrangiano aumentado pode ser escrito como

$$\begin{aligned} L(\mathbf{E}, \mathbf{B}, \mathbf{Z}, \mathbf{U}) = & \tau \|\mathbf{E}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} \\ & + \frac{\rho}{2} \|\tilde{\mathbf{A}}\mathbf{E} + \tilde{\mathbf{B}}\mathbf{B} + \tilde{\mathbf{C}}\mathbf{Z} + \tilde{\mathbf{D}} + \mathbf{U}\|_F^2 - \frac{\rho}{2} \|\mathbf{U}\|_F^2, \end{aligned} \quad (3.26)$$

onde $\mathbf{U} = (1/\rho)\mathbf{Y}$ é a variável dual escalada. Note que ainda é possível escrever Equação 3.26 de outra forma. Sejam \mathbf{V}_1 e \mathbf{V}_2 duas matrizes quaisquer de dimensões $v_1 \times s$ e

³ Considera-se que quando há ocorrência de um *outlier* todas as saídas são afetadas simultaneamente.

$v_2 \times s$, respectivamente. Seja ainda \mathbf{V} uma matriz de dimensões $(v_1 + v_2) \times s$ formada pela concatenação de \mathbf{V}_1 e \mathbf{V}_2 , da seguinte forma: $\mathbf{V} = [\mathbf{V}_1^\top \quad \mathbf{V}_2^\top]^\top$. Então,

$$\begin{aligned} \|\mathbf{V}\|_F^2 &= \sum_{i=1}^{v_1+v_2} \sum_{j=1}^s |v_{ij}|^2 \\ &= \sum_{i=1}^{v_1} \sum_{j=1}^s |v_{ij}|^2 + \sum_{i=(v_1+1)}^{v_1+v_2} \sum_{j=1}^s |v_{ij}|^2 \\ &= \|\mathbf{V}_1\|_F^2 + \|\mathbf{V}_2\|_F^2. \end{aligned} \quad (3.27)$$

Como $\widetilde{\mathbf{A}}\mathbf{E} + \widetilde{\mathbf{B}}\mathbf{B} + \widetilde{\mathbf{C}}\mathbf{Z} + \widetilde{\mathbf{D}} + \mathbf{U} = [(\mathbf{H}\mathbf{B} - \mathbf{T} - \mathbf{E} + \mathbf{U}_1)^\top \quad (\mathbf{B} - \mathbf{Z} + \mathbf{U}_2)^\top]^\top$, onde $\mathbf{U} = [\mathbf{U}_1^\top \quad \mathbf{U}_2^\top]^\top$, com $\mathbf{U}_1 \in \mathbb{R}^{N \times m}$ e $\mathbf{U}_2 \in \mathbb{R}^{\tilde{N} \times m}$, segue-se que o Lagrangiano aumentado (Equação 3.26) pode ser escrito como

$$\begin{aligned} L(\mathbf{E}, \mathbf{B}, \mathbf{Z}, \mathbf{U}) &= \tau \|\mathbf{E}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \lambda_1 \|\mathbf{Z}\|_{2,1} \\ &\quad + \frac{\rho}{2} \|\mathbf{H}\mathbf{B} - \mathbf{T} - \mathbf{E} + \mathbf{U}_1\|_F^2 + \frac{\rho}{2} \|\mathbf{B} - \mathbf{Z} + \mathbf{U}_2\|_F^2 \\ &\quad - \frac{\rho}{2} \|\mathbf{U}_1\|_F^2 - \frac{\rho}{2} \|\mathbf{U}_2\|_F^2. \end{aligned} \quad (3.28)$$

Na iteração k , o ADMM para solução do GOR-ELM consiste nas seguintes regras de atualização para:

1. \mathbf{B}^{k+1} , tem-se o seguinte subproblema

$$\mathbf{B}^{k+1} := \arg \min_{\mathbf{B}} \frac{\rho}{2} \|\mathbf{H}\mathbf{B} - \mathbf{T} - \mathbf{E}^k + \mathbf{U}_1^k\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \frac{\rho}{2} \|\mathbf{B} - \mathbf{Z}^k + \mathbf{U}_2^k\|_F^2. \quad (3.29)$$

Fazendo o gradiente da função objetivo com respeito a \mathbf{B} igual a $\mathbf{0}$, tem-se

$$\mathbf{H}^\top (\mathbf{H}\mathbf{B} - (\mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k)) + \frac{\lambda_2}{\rho} \mathbf{B} + (\mathbf{B} - \mathbf{Z}^k + \mathbf{U}_2^k) = \mathbf{0}, \quad (3.30)$$

e a solução para Equação 3.29 é

$$\mathbf{B} = (\mathbf{H}^\top \mathbf{H} + \eta \mathbf{I})^{-1} [\mathbf{H}^\top (\mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k) + (\mathbf{Z}^k - \mathbf{U}_2^k)], \quad (3.31)$$

onde $\eta = (\lambda_2 + \rho)/\rho$.

2. \mathbf{Z}^{k+1} , tem-se o seguinte subproblema

$$\mathbf{Z}^{k+1} := \arg \min_{\mathbf{Z}} \frac{\lambda_1}{\rho} \|\mathbf{Z}\|_{2,1} + \frac{1}{2} \|\mathbf{B}^{k+1} - \mathbf{Z} + \mathbf{U}_2^k\|_F^2. \quad (3.32)$$

Note que esse problema de otimização é idêntico ao estabelecido em Equação 3.9. Logo, a solução para Equação 3.32 é dada por

$$\mathbf{Z}^{k+1} = \mathcal{S}_{\frac{\lambda_1}{\rho}} (\mathbf{B}^{k+1} + \mathbf{U}_2^k). \quad (3.33)$$

3. \mathbf{E}^{k+1} , tem-se o seguinte subproblema

$$\mathbf{E}^{k+1} := \arg \min_{\mathbf{E}} \frac{\tau}{\rho} \|\mathbf{E}\|_{2,1} + \frac{1}{2} \left\| \mathbf{H}\mathbf{B}^{k+1} - \mathbf{T} - \mathbf{E} + \mathbf{U}_1^k \right\|_F^2, \quad (3.34)$$

que possui a mesma estrutura de Equação 3.9 e Equação 3.32, e portanto possui a seguinte solução

$$\mathbf{E}^{k+1} = \mathcal{S}_{\frac{\tau}{\rho}} \left(\mathbf{H}\mathbf{B}^{k+1} - \mathbf{T} + \mathbf{U}_1^k \right). \quad (3.35)$$

4. \mathbf{U}^{k+1} , o multiplicador de Lagrange é atualizado por

$$\mathbf{U}_1^{k+1} := \mathbf{U}_1^k + \left(\mathbf{H}\mathbf{B}^{k+1} - \mathbf{T} - \mathbf{E}^{k+1} \right) \quad (3.36)$$

$$\mathbf{U}_2^{k+1} := \mathbf{U}_2^k + \left(\mathbf{B}^{k+1} - \mathbf{Z}^{k+1} \right). \quad (3.37)$$

O algoritmo 3 resume o GOR-ELM.

Algoritmo 3: GOR-ELM: Generalized Outlier Robust Extreme Learning Machine

- 1 **Requer:** Amostras de treino $(\mathbf{x}_p, \mathbf{t}_p)$, com $p = 1, \dots, N$, função de ativação $h(\cdot)$, e parâmetros de regularização λ_1 , λ_2 , τ , e ρ .
 - 2 **Inicializar:** \mathbf{B}^0 , \mathbf{U}^0 , \mathbf{Z}^0 , \mathbf{E}^0 , e $k = 0$.
 - 3 Atribuir os parâmetros \mathbf{A} e $\boldsymbol{\nu}$ aleatoriamente.
 - 4 Calcular a matriz de saída da camada oculta \mathbf{H} usando Equação 2.34.
 - 5 **repita**
 - 6 $\mathbf{B}^{k+1} := \arg \min_{\mathbf{B}} \frac{\rho}{2} \left\| \mathbf{H}\mathbf{B} - \mathbf{T} - \mathbf{E}^k + \mathbf{U}_1^k \right\|_F^2 + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2 + \frac{\rho}{2} \left\| \mathbf{B} - \mathbf{Z}^k + \mathbf{U}_2^k \right\|_F^2$
 - 7 $\mathbf{Z}^{k+1} := \arg \min_{\mathbf{Z}} \frac{\lambda_1}{\rho} \|\mathbf{Z}\|_{2,1} + \frac{1}{2} \left\| \mathbf{B}^{k+1} - \mathbf{Z} + \mathbf{U}_2^k \right\|_F^2$
 - 8 $\mathbf{E}^{k+1} := \arg \min_{\mathbf{E}} \frac{\tau}{\rho} \|\mathbf{E}\|_{2,1} + \frac{1}{2} \left\| \mathbf{H}\mathbf{B}^{k+1} - \mathbf{T} - \mathbf{E} + \mathbf{U}_1^k \right\|_F^2$
 - 9 $\mathbf{U}_1^{k+1} := \mathbf{U}_1^k + \left(\mathbf{H}\mathbf{B}^{k+1} - \mathbf{T} - \mathbf{E}^{k+1} \right)$
 - 10 $\mathbf{U}_2^{k+1} := \mathbf{U}_2^k + \left(\mathbf{B}^{k+1} - \mathbf{Z}^{k+1} \right)$
 - 11 $k = k + 1$
 - 12 **até** atender o critério de parada.
-

3.4 Nota computacional

A atualização de \mathbf{B} estabelecida na Equação 3.8 para o GR-ELM e na Equação 3.31 para o GOR-ELM envolve a inversa de $\mathbf{G} = \left(\mathbf{H}^\top \mathbf{H} + \eta \mathbf{I} \right)$ que é feita usando a fatoração de Cholesky seguida de substituições *forward* e *backward*. Mais detalhes sobre a fatoração de Cholesky e substituições *forward* e *backward* são dados nos Apêndice D e Apêndice E, respectivamente. Como mantém-se ρ fixo, é possível fatorar \mathbf{G} apenas uma vez, e utilizar a fatoração armazenada nas etapas subsequentes.

Conforme Equação 2.34, em que \mathbf{H} é uma matriz $\mathbb{R}^{N \times \widetilde{N}}$, quando $\widetilde{N} > N$ pode-se usar o lema de inversão de matrizes para reduzir o custo computacional

$$\begin{aligned} (\eta \mathbf{I} + \mathbf{H}^\top \mathbf{H})^{-1} &= (\eta \mathbf{I})^{-1} - (\eta \mathbf{I})^{-1} \mathbf{H}^\top (\mathbf{I}^{-1} + \mathbf{H} (\eta \mathbf{I})^{-1} \mathbf{H}^\top)^{-1} \mathbf{H} (\eta \mathbf{I})^{-1} \\ &= \frac{1}{\eta} \left(\mathbf{I} - \mathbf{H}^\top (\eta \mathbf{I} + \mathbf{H} \mathbf{H}^\top)^{-1} \mathbf{H} \right). \end{aligned} \quad (3.38)$$

Então, a fatoração de $(\eta \mathbf{I} + \mathbf{H} \mathbf{H}^\top)$ pode ser armazenada e substituições *forward* e *backward* podem ser usadas na atualização. Esse procedimento também aplica-se à Equação 3.20.

A eficiência computacional do GR-ELM é afetada basicamente pelas atualizações de \mathbf{B} e \mathbf{Z} , e no GOR-ELM, além das atualizações de \mathbf{B} e \mathbf{Z} , tem-se a atualização de \mathbf{E} . Como a atualização de \mathbf{Z} (e de \mathbf{E}) envolve apenas o operador *block soft-thresholding*, o tempo computacional para resolver a inversa de \mathbf{G} domina o tempo computacional total de uma iteração. Logo, ao armazenar a decomposição de Cholesky de \mathbf{G} e usar as substituições *forward* e *backward* nas atualizações seguintes, obtém-se uma considerável melhoria no custo computacional total dos métodos propostos.

3.5 Critério de Parada

Neste trabalho, segue-se a metodologia sugerida por Boyd (2010) para o critério de parada do GR-ELM. O algoritmo GR-ELM termina quando os resíduos primal e dual, dados por

$$\mathbf{R}^k = \mathbf{B}^k - \mathbf{Z}^k \text{ e } \mathbf{S}^k = \rho(\mathbf{Z}^k - \mathbf{Z}^{k-1}), \quad (3.39)$$

respectivamente, satisfazem

$$\|\mathbf{R}^k\|_F \leq \epsilon^{\text{pri}} \quad \text{e} \quad \|\mathbf{S}^k\|_F \leq \epsilon^{\text{dual}}. \quad (3.40)$$

As tolerâncias $\epsilon^{\text{pri}} > 0$ e $\epsilon^{\text{dual}} > 0$ são obtidas usando o critério absoluto e relativo, conforme sugerido por (BOYD, 2010),

$$\epsilon^{\text{pri}} = \sqrt{m\widetilde{N}} \epsilon^{\text{abs}} + \epsilon^{\text{rel}} \max \left\{ \|\mathbf{B}^k\|_F, \|\mathbf{Z}^k\|_F \right\}, \quad (3.41)$$

$$\epsilon^{\text{dual}} = \sqrt{m\widetilde{N}} \epsilon^{\text{abs}} + \epsilon^{\text{rel}} \rho \|\mathbf{U}^k\|_F, \quad (3.42)$$

onde $\epsilon^{\text{abs}} > 0$ e $\epsilon^{\text{rel}} > 0$ são chamadas de tolerâncias absoluta e relativa, respectivamente.

Note que o problema estabelecido para o GR-ELM (Equação 3.3) segue a estrutura do ADMM dado pela Equação 2.22 (também chamado de ADMM de 2 blocos de variáveis) e portanto converge globalmente para qualquer $\rho > 0$. Além disso, o critério de parada para esse problema é bem estabelecido na literatura. Para os problemas ADMM de 3 blocos de variáveis, como é o caso do GOR-ELM, a convergência não é garantida e os critérios de

paradas para esses algoritmos se baseiam apenas no número de iterações. Recentemente, Chen et al. (2014) mostraram que problemas ADMM de 3 blocos possuem convergência garantida quando $\widetilde{\mathbf{A}}^\top \widetilde{\mathbf{B}} = \mathbf{0}$, $\widetilde{\mathbf{A}}^\top \widetilde{\mathbf{C}} = \mathbf{0}$ ou $\widetilde{\mathbf{B}}^\top \widetilde{\mathbf{C}} = \mathbf{0}$. Para o GOR-ELM, tem-se que $\widetilde{\mathbf{A}}^\top \widetilde{\mathbf{C}} = \mathbf{0}$ e portanto o algoritmo converge. Como critério de parada para o GOR-ELM, utiliza-se um número fixo de iterações.

3.6 Generalizações

A proposta deste capítulo traz no GR-ELM uma generalização do R-ELM. Se $\mathbf{T} = \mathbf{y} \in \mathbb{R}^m$, então a Equação 3.2, aqui repetida por conveniência,

$$\underset{\mathbf{B}}{\text{minimize}} \quad \frac{C}{2} \|\mathbf{H}\mathbf{B} - \mathbf{T}\|_F^2 + \lambda_1 \|\mathbf{B}\|_{2,1} + \frac{\lambda_2}{2} \|\mathbf{B}\|_F^2, \quad (3.43)$$

se resume ao problema de regularização *elastic net* (Equação 2.39). Além disso, quando $\lambda_1 = 0$ e $C = 1$ tem-se a regularização *ridge* para múltiplas saídas com parâmetro *ridge* em comum (λ_2) e a solução é uma extensão da Equação 2.41 dada por

$$\mathbf{B} = \left(\mathbf{H}^\top \mathbf{H} + \lambda_2 \mathbf{I} \right)^{-1} \mathbf{H}^\top \mathbf{T}. \quad (3.44)$$

Na Equação 3.43 é necessário o uso apenas dos parâmetros de regularização λ_1 e λ_2 de tal forma que o problema de minimização resultante com os dois parâmetros é equivalente ao problema original (Equação 3.2). Entretanto, ao aplicar o ADMM para resolver a Equação 3.2, a atualização de \mathbf{B}^{k+1} (Equação 3.6) depende de $\eta = (\lambda_2 + \rho)/C$, que seria limitada se $C = 1$, já que usualmente $\rho = 1$ é fixo. Se $C = 1$ então η será sempre maior que um. A única forma de obter $\eta < 1$ com $C = 1$ é mudando simultaneamente λ_2 e ρ , que neste trabalho é evitado já que ρ também é utilizado na atualização de \mathbf{Z}^{k+1} (Equação 3.13). Assim, ao manter C no problema tem-se um maior controle do elemento que será adicionado à diagonal de $\mathbf{H}^\top \mathbf{H}$, e consequentemente na estabilidade da inversa.

Outra proposta apresentada neste capítulo foi o GOR-ELM que é uma generalização do OR-ELM. Se, $\mathbf{T} = \mathbf{y} \in \mathbb{R}^m$ e $\lambda_1 = 0$, então a Equação 3.22 é equivalente ao problema do OR-ELM (Equação 2.43). Ao fazer $\lambda_1 > 0$, é possível obter uma rede mais compacta mantendo-se a robustez à *outliers*. Note que diferente do caso do GR-ELM, no GOR-ELM $\eta = (\lambda_2 + \rho)/\rho \geq 1$ já que ρ é fixo e igual a um. Nos testes realizados neste trabalho isso não foi um problema.

4 Experimentos de Classificação Multiclasse

Neste capítulo, avalia-se o GR-ELM proposto na seção 3.1. 14 bancos de dados de classificação binária (Tabela 1) e oito bases de dados de classificação multiclasse (Tabela 2) são selecionados para realização dos testes. Esses dados foram retirados do repositório UCI e do portal LIBSVM (ASUNCION; NEWMAN, 2007; CHANG; LIN, 2011).

Tabela 1 – Informações sobre o banco de dados de classificação binária.

Banco	# Dados de treino	# Dados de teste	# Atributos
Bupa	173	172	6
Australia	346	344	14
Breast Cancer	342	341	10
Diabetes	384	384	8
Heart	135	135	13
Ionosphere	176	175	34
Mushroom	4062	4062	22
SVMGuide1	3089	3089	4
Magic	9510	9510	11
COD RNA	29768	29767	8
Colon Cancer	31	31	2000
Leukemia	38	34	7129
Spambase	2301	2300	57
Adult	6414	26147	123

Tabela 2 – Informação sobre o banco de dados de classificação multiclasse.

Banco	# Dados de treino	# Dados de teste	# Atributos	# Classes
Iris	75	75	4	3
Wine	90	88	13	3
Vowel	528	462	10	11
Segment	1155	1155	19	7
Satimage	4435	2000	36	6
DNA	2000	1186	180	3
SVMGuide2	197	194	20	3
USPS	7291	2007	256	10

Para os bancos de dados de classificação binária, as saídas (*labels*) são -1 ou 1. Para os dados de classificação multiclasse com m classes, a k -ésima classe é representada por um vetor m -dimensional com todos os elementos iguais a -1 exceto pelo k -ésimo elemento do vetor, que é igual a 1. Os atributos dos dados de treino são normalizados para terem valores entre $[-1, 1]$. Os dados de teste são normalizados de acordo com os fatores usados na normalização dos dados de treino.

Todos os experimentos foram conduzidos em um computador com 3.6 GHz core i7 e 8 GB de RAM. Simulações do GR-ELM são realizadas no MATLAB 8.5 e as simulações do DGR-ELM são feitas em C usando MPI e GNU *Scientific Library* (GSL) para álgebra linear com a biblioteca ATLAS.

4.1 Especificações dos Parâmetros

O número inicial de neurônios, \tilde{N} , na camada oculta é de 1000 para todos os bancos de dados. Para o ADMM, fixa-se o $\rho = 1$ e os parâmetros de regularização λ_1 e C são selecionados por validação cruzada com 5-fold. Testou-se 14 valores para C : $[0.01, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000]$. Para λ_1 , 100 valores decrescentes foram testados variando-os de 100 a λ_{\min} em escala logarítmica. O valor de λ_{\min} é definido como 0.0001 se $N > n$ e 0.01 caso contrário. O parâmetro λ_2 é fixado em 0.1. Utiliza-se os mesmos parâmetros tanto no GR-ELM quanto no DGR-ELM. Os valores de \mathbf{A} e $\boldsymbol{\nu}$ são gerado aleatoriamente baseados em uma distribuição uniforme e a função $h(\mathbf{x}_k; \mathbf{a}_i, \nu_i) = 1/(1 + \exp(-\mathbf{a}_i \cdot \mathbf{x} + \nu_i))$ é escolhida como função de ativação para o GR-ELM e DGR-ELM. Para o GR-ELM, a tolerância absoluta e relativa são $\epsilon^{\text{abs}} = 10^{-3}$ e $\epsilon^{\text{rel}} = 10^{-2}$, respectivamente. Por conveniência e simplificação na codificação, o número de iterações do DGR-ELM é fixo em 100.

4.2 Avaliação do GR-ELM

Nesta seção, compara-se o GR-ELM e o ELM com regularização *ridge*, que daqui por diante será representado por R-ELM¹. Os bancos de dados são divididos em dois conjuntos de tamanhos aproximadamente iguais: um de treino e outro de teste. A divisão dos dados é feita de tal forma que os conjuntos de treino e teste possuam, aproximadamente, a mesma proporção do número de amostras por classe. O conjunto de treino é usado para obter os parâmetros e os pesos de saída e o conjunto de teste é usado para avaliar a rede treinada em dados que não foram usados na etapa de treinamento.

As Tabelas 3 e 4 resumem a acurácia média de treino e teste (e os desvios padrões correspondentes) e os tempos médio para os bancos de dados de classificação binária e multiclasse, respectivamente, em 20 execuções do experimento para cada banco de dados. O experimento é definido da seguinte forma: novos valores de \mathbf{A} e $\boldsymbol{\nu}$ são gerados; realiza-se uma permutação aleatória dentro do conjunto de treino e teste; a rede neural é treinada usando o GR-ELM e o R-ELM; e as métricas de avaliação são obtidas. A acurácia foi utilizada como métrica para avaliação dos métodos nos bancos de dados de classificação multiclasse e binária. Embora outras métricas possam ser empregadas para avaliação dos

¹ Note que o ELM com regularização *ridge* é um caso especial do R-ELM, quando $\lambda_1 = 0$.

métodos em bancos de dados de classificação multiclasse, nesta Tese, optou-se pelo uso da acurácia, comumente encontrada na literatura de ELM para tarefas de classificação multiclasse (HUANG et al., 2012; CHOROWSKI; WANG; ZURADA, 2014; WANG et al., 2016; BAI et al., 2014).

A Tabela 5 resume os parâmetros de regularização obtidos na validação cruzada com 5-*fold*. Neste caso, foi definido uma validação cruzada com 5-*fold* para diminuir o custo computacional dos experimentos e atender às bases de dados que possuem um número reduzido de amostras de treino. Para a divisão dos *folds*, matém-se a proporção do número de amostras por classe em cada *fold*. A Tabela 6 exhibe o número médio de neurônios em 20 execuções para os bancos de dados de classificação binária e multiclasse.

Tabela 3 – Acurácia média com os respectivos desvios padrões e tempo de execução para os bancos de dados de classificação binária ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GR-ELM				R-ELM			
	Ac. Treino	Ac. Teste	Tempo Treino	Tempo Teste	Ac. Treino	Ac. Teste	Tempo Treino	Tempo Teste
Bupa	80.46 ± 0.81	71.77 \pm 0.89	0.0107	0.0022	81.07 ± 0.67	71.42 ± 0.95	0.0008	0.0025
Australian	88.73 ± 0.48	84.59 \pm 0.46	0.0137	0.0006	88.53 ± 0.35	84.52 ± 0.39	0.0025	0.0053
Breast Cancer	97.50 ± 0.20	96.60 ± 0.26	0.0170	0.0005	97.87 ± 0.19	96.74 \pm 0.09	0.0023	0.0050
Diabetes	79.02 ± 0.43	76.85 ± 0.55	0.0490	0.0005	79.93 ± 0.35	76.86 \pm 0.37	0.0033	0.0058
Heart	87.63 ± 0.35	87.26 \pm 0.70	0.0078	0.0008	87.41 ± 0.00	87.15 ± 0.69	0.0007	0.0022
Ionosphere	96.14 ± 0.57	87.29 \pm 0.67	0.0096	0.0008	97.05 ± 0.40	87.14 ± 0.57	0.0010	0.0028
Mushroom	100.00 ± 0.00	100.00 \pm 0.00	0.1049	0.0835	100.00 ± 0.00	100.00 \pm 0.00	0.0403	0.0822
SVMGuide	97.14 ± 0.05	96.74 \pm 0.04	0.1556	0.0854	97.09 ± 0.05	96.67 ± 0.05	0.0392	0.0817
Magic	88.00 ± 0.11	86.63 \pm 0.14	0.1632	0.1424	87.87 ± 0.09	86.63 \pm 0.10	0.0993	0.1841
COD RNA	95.68 ± 0.04	95.23 \pm 0.05	0.4347	0.6348	95.50 ± 0.05	95.20 ± 0.06	0.3435	0.6553
Colon Cancer	100.00 ± 0.00	82.90 \pm 4.07	0.0052	0.0015	100.00 ± 0.00	82.42 ± 4.25	0.0001	0.0022
Leukemia	100.00 ± 0.00	76.91 \pm 4.80	0.0050	0.0046	100.00 ± 0.00	76.76 ± 4.47	0.0001	0.0059
Spambase	95.49 ± 0.17	93.09 \pm 0.20	0.1038	0.0452	95.08 ± 0.12	92.99 ± 0.21	0.0389	0.0529
Adult	85.20 ± 0.18	84.18 ± 0.08	0.1302	0.3905	84.98 ± 0.15	84.21 \pm 0.06	0.0622	0.5340

Tabela 4 – Acurácia média com os respectivos desvios padrões e tempos de execução para bancos de dados de classificação multiclasse ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GR-ELM				R-ELM			
	Ac. Treino	Ac. Teste	Tempo Treino	Tempo Teste	Ac. Treino	Ac. Teste	Tempo Treino	Tempo Teste
Iris	100.0 \pm 0.00	94.53 \pm 0.96	0.0117	0.0004	98.67 \pm 0.00	93.07 \pm 1.02	0.0003	0.0014
Wine	100.0 \pm 0.00	95.57 \pm 0.82	0.0106	0.0015	100.0 \pm 0.00	95.57 \pm 0.82	0.0005	0.0015
Vowel	100.0 \pm 0.00	50.02 \pm 1.93	0.2739	0.0072	100.0 \pm 0.00	50.18 \pm 1.97	0.0061	0.0073
Segment	97.82 \pm 0.14	95.75 \pm 0.26	0.4296	0.0200	97.98 \pm 0.14	95.74 \pm 0.26	0.0252	0.0210
Satimage	93.52 \pm 0.14	90.26 \pm 0.32	0.3359	0.0459	93.86 \pm 0.15	90.35 \pm 0.29	0.0439	0.0426
DNA	97.99 \pm 0.18	93.38 \pm 0.55	0.2079	0.0267	98.03 \pm 0.17	93.36 \pm 0.56	0.0288	0.0267
SVMGuide2	92.28 \pm 0.80	81.96 \pm 0.73	0.0162	0.0008	93.68 \pm 0.31	81.34 \pm 0.88	0.0010	0.0031
USPS	97.87 \pm 0.12	92.21 \pm 0.25	0.2646	0.0535	97.95 \pm 0.11	92.26 \pm 0.22	0.0713	0.0522

Tabela 5 – Especificações de parâmetros dos bancos de dados binários e o número médio de neurônios ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GR-ELM			R-ELM	
	λ_1	C	Núm. Neur.	C	Núm. Neur.
Bupa	0.0201	5	803.80	50	1000
Australian	1.3219	10	66.95	5	1000
Breast Cancer	0.5722	5	48.50	10	1000
Diabetes	6.1359	50	23.95	10	1000
Heart	0.0534	0.5	256.90	1	1000
Ionosphere	0.1630	2	217.40	5	1000
Mushroom	0.1630	500	950.05	1000	1000
SVMGuide	0.0001	1000	1000.0	1000	1000
Magic	0.5722	50	778.55	200	1000
COD RNA	1.1498	1000	963.45	1000	1000
Colon Cancer	0.0145	2	642.45	10	1000
Leukemia	0.0100	1000	742.20	1000	1000
Spambase	0.0351	2	880.20	10	1000
Adult	0.0201	0.1	732.40	0.5	1000

Tabela 6 – Especificações de parâmetros para os bancos de dados multiclasse e número médio de neurônios ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GR-ELM			R-ELM	
	λ_1	C	Núm. Neur.	C	Núm. Neur.
Iris	2.3101	1000	186.90	1000	1000
Wine	0.0001	50	1000.0	500	1000
Vowel	0.0001	1000	1000.0	1000	1000
Segment	0.1630	100	999.65	1000	1000
Satimage	0.1630	10	999.55	100	1000
DNA	0.0038	0.2	999.90	2	1000
SVMGuide2	0.5722	5	193.00	10	1000
USPS	0.2154	1	1000.0	5	1000

As simulações indicam que o GR-ELM possui acurácia de teste similar quando comparada ao ELM com regularização *ridge*, embora geralmente um tempo menor de teste e uma rede mais compacta é obtida com o GR-ELM. O tempo de treino para o GR-ELM é maior que o do R-ELM, que é esperado já que resolver a Equação 3.2 é computacionalmente mais custoso do que o ELM com regularização *ridge*. Entretanto, não há necessidade de saber a priori o número de neurônios quando se trabalha com o GR-ELM. Se o número de neurônios for suficientemente grande, o ELM com regularização *ridge* e o GR-ELM terão desempenhos similares sob a ótica da acurácia de teste. Contudo, um menor tempo de execução para o teste e uma rede mais compacta são esperados para o GR-ELM.

No intuito de se obter um critério rigoroso nas análises, realizou-se testes estatísticos para verificar se houve diferença significativa entre os métodos GR-ELM e R-ELM. O teste aplicado foi o *Wilcoxon Signed Rank*. A hipótese nula, H_0 , para o teste é que os dois métodos têm desempenho igual. Considerando um nível de significância de 0.05, têm-se

- para a acurácia de treino, não há evidências suficientes para rejeitar a hipótese nula (valor- $p = 0.4533$, hipótese alternativa é que as acurácias são diferentes);
- para a acurácia de teste, com um valor- p de 0.03534, rejeita-se H_0 em favor de H_1 : o GR-ELM possui acurácia maior de teste do que o R-ELM;
- para o tempo de treino, com valor- p menor que 0.001, rejeita-se H_0 em favor de H_1 : o tempo de treino do GR-ELM é maior do que o tempo de treino do R-ELM; e
- para o tempo de teste, com um valor- p de 0.01276, rejeita-se H_0 , em favor de H_1 : o GR-ELM possui um tempo de teste menor que o R-ELM

Assim, pelos testes estatísticos realizados, têm-se que o GR-ELM, como esperado, possui um tempo de treino maior que o R-ELM. Além disso, a acurácia de treino obtida no GR-ELM foi compatível com a do R-ELM. Contudo, a acurácia de teste do GR-ELM se mostrou estatisticamente maior que a do R-ELM. Já para o tempo de teste, o GR-ELM demonstrou ter um tempo de teste estatisticamente menor que o tempo de teste do R-ELM.

4.3 Avaliação do DGR-ELM

Nesta seção, avalia-se a versão distribuída do GR-ELM. Embora o DGR-ELM possua uma aplicação mais atrativa para bancos de dados grandes, utiliza-se todos os bancos de dados fornecidos nos experimentos do GR-ELM para avaliar o DGR-ELM, e verificar seu comportamento em bancos de dados menores. A Tabela 7 resume a acurácia média de 20 execuções de um experimento para os problemas de classificação binária usando o DGR-ELM com 1, 2 e 4 processadores. O mesmo foi feito para os bancos de dados de classificação multiclasse e os resultados são apresentados na Tabela 8. O experimento é definido de forma equivalente ao feito na avaliação do GR-ELM.

Como esperado, as acurácias tanto para o treino quanto para o teste nos problemas de classificação binária e multiclasse são similares quando comparadas com as obtidas pelo GR-ELM. Em outras palavras, os dados de treino podem estar distribuídos em diferentes máquinas e o DGR-ELM pode obter acurácia de treino e teste similar ao caso quando todo o conjunto de treino está concentrado em uma máquina apenas.

A Tabela 9 resume o tempo médio de treino e teste para as 20 execuções do experimento com DGR-ELM com 1, 2 e 4 processadores para os problemas de classificação

Tabela 7 – Acurácia média com os respectivos desvios padrões para treino e teste nos problemas de classificação binária usando o DGR-ELM com 1, 2 e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	1		2		4	
	Ac. Treino	Ac. Teste	Ac. Treino	Ac. Teste	Ac. Treino	Ac. Teste
Bupa	79.95 \pm 0.55	72.66 \pm 0.81	78.92 \pm 0.49	72.34 \pm 0.29	77.46 \pm 0.00	72.09 \pm 0.00
Australian	89.25 \pm 0.42	84.80 \pm 0.54	88.73 \pm 0.23	84.40 \pm 0.50	88.17 \pm 0.37	84.82 \pm 0.12
Breast Cancer	97.61 \pm 0.21	96.91 \pm 0.23	97.46 \pm 0.14	96.60 \pm 0.17	97.47 \pm 0.14	96.63 \pm 0.15
Diabetes	78.98 \pm 0.55	76.86 \pm 0.44	79.06 \pm 0.19	77.04 \pm 0.33	78.48 \pm 0.13	77.17 \pm 0.70
Heart	88.01 \pm 0.28	87.59 \pm 0.55	87.39 \pm 0.10	87.42 \pm 0.10	87.41 \pm 0.00	87.21 \pm 0.32
Ionosphere	96.41 \pm 0.42	87.45 \pm 0.86	94.73 \pm 0.26	87.02 \pm 0.26	93.20 \pm 0.80	86.49 \pm 0.53
Mushroom	100.0 \pm 0.00	100.0 \pm 0.00	100.0 \pm 0.00	100.0 \pm 0.00	100.0 \pm 0.00	100.0 \pm 0.00
SVMGuide	97.13 \pm 0.04	96.73 \pm 0.04	97.10 \pm 0.05	96.76 \pm 0.05	97.09 \pm 0.05	96.77 \pm 0.04
Magic	87.95 \pm 0.11	86.63 \pm 0.10	87.77 \pm 0.09	86.62 \pm 0.10	87.52 \pm 0.09	86.51 \pm 0.10
COD RNA	95.37 \pm 0.06	95.13 \pm 0.05	95.30 \pm 0.05	95.10 \pm 0.05	95.24 \pm 0.05	95.07 \pm 0.05
Colon Cancer	100.0 \pm 0.00	80.84 \pm 2.18	100.0 \pm 0.00	79.42 \pm 3.15	100.0 \pm 0.00	80.77 \pm 2.66
Leukemia	100.0 \pm 0.00	77.88 \pm 4.09	100.0 \pm 0.00	78.53 \pm 3.24	100.0 \pm 0.00	78.18 \pm 4.60
Spambase	95.45 \pm 0.18	93.06 \pm 0.21	94.90 \pm 0.20	93.05 \pm 0.19	94.18 \pm 0.13	92.96 \pm 0.21
Adult	85.21 \pm 0.15	84.17 \pm 0.09	84.73 \pm 0.11	84.16 \pm 0.07	84.27 \pm 0.14	84.09 \pm 0.07

Tabela 8 – Acurácia média com os respectivos desvios padrões nos problemas de classificação multiclasse usando o DGR-ELM com 1, 2 e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	1		2		4	
	Ac. Treino	Ac. Teste	Ac. Treino	Ac. Teste	Ac. Treino	Ac. Teste
Iris	100.0 \pm 0.00	94.64 \pm 0.19	100.0 \pm 0.00	93.33 \pm 0.00	98.67 \pm 0.00	93.33 \pm 0.00
Wine	100.0 \pm 0.00	95.45 \pm 0.00	100.0 \pm 0.00	96.57 \pm 0.16	100.0 \pm 0.00	96.59 \pm 0.00
Vowel	100.0 \pm 0.00	50.42 \pm 1.78	100.0 \pm 0.00	48.76 \pm 1.96	100.0 \pm 0.00	49.71 \pm 1.60
Segment	97.89 \pm 0.13	95.77 \pm 0.24	97.31 \pm 0.16	95.54 \pm 0.22	96.74 \pm 0.13	95.00 \pm 0.23
Satimage	93.53 \pm 0.16	90.29 \pm 0.25	92.75 \pm 0.17	89.91 \pm 0.25	91.89 \pm 0.15	89.46 \pm 0.25
DNA	97.90 \pm 0.23	93.21 \pm 0.40	97.33 \pm 0.22	93.25 \pm 0.38	96.34 \pm 0.30	92.67 \pm 0.63
SVMGuide2	91.30 \pm 0.74	81.44 \pm 0.34	91.21 \pm 0.24	81.94 \pm 0.42	89.73 \pm 1.10	82.11 \pm 0.76
USPS	97.86 \pm 0.13	92.20 \pm 0.37	97.56 \pm 0.10	92.11 \pm 0.30	97.15 \pm 0.11	91.86 \pm 0.35

binária. Os mesmos resultados para os problemas de classificação multiclasse são mostrados na Tabela 10. Uma observação importante é que não faz parte do objetivo deste trabalho diminuir o tempo de treino. Menores tempos de treino podem ser obtidos usando LAPACK ao invés de GSL para a fatoração de Cholesky, e através do uso de pacotes de álgebra linear baseados em GPU como sugerido por Boyd (2010).

Tabela 9 – Tempo médio para problemas de classificação binária usando o DGR-ELM com 1, 2, e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	1		2		4	
	Time Treino	Tempo Teste	Tempo Treino	Tempo Teste	Tempo Treino	Tempo Teste
Bupa	0.0408	0.0055	0.0159	0.0062	0.0169	0.007
Australian	0.0875	0.0009	0.0414	0.0011	0.025	0.002
Breast Cancer	0.0855	0.0006	0.046	0.0011	0.0244	0.0017
Diabetes	0.1015	0.0004	0.0489	0.0005	0.0361	0.0006
Heart	0.0334	0.0013	0.0135	0.0021	0.015	0.0032
Ionosphere	0.0416	0.0016	0.0175	0.0025	0.0175	0.0036
Mushroom	0.486	0.1978	0.4132	0.2033	0.5163	0.2141
SVMGuide	0.4165	0.1926	0.3801	0.1961	0.6586	0.2004
Magic	0.8767	0.3986	0.6108	0.4148	0.657	0.4538
COD RNA	2.3304	1.1616	1.4491	1.3217	1.1825	1.4507
Colon Cancer	0.008	0.008	0.008	0.0104	0.01	0.0118
Leukemia	0.0087	0.0321	0.008	0.0386	0.0096	0.0426
Spambase	0.3597	0.1066	0.3446	0.1127	0.4251	0.1188
Adult	0.6548	1.3012	0.4992	1.4524	0.5684	1.6618

Tabela 10 – Tempo médio para problemas de classificação multiclasse usando o DGR-ELM com 1, 2, e 4 processadores ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	1		2		4	
	Tempo Treino	Tempo Teste	Tempo Treino	Tempo Teste	Tempo Treino	Tempo Teste
Iris	0.0324	0.0006	0.024	0.0009	0.0216	0.0015
Wine	0.0401	0.0038	0.0303	0.0041	0.0234	0.0045
Vowel	0.4905	0.0202	0.2442	0.021	0.1877	0.0218
Segment	0.5817	0.0577	0.5156	0.0578	0.3324	0.06
Satimage	0.766	0.1029	0.8923	0.1051	1.4223	0.1091
DNA	0.4373	0.0794	0.5015	0.0818	0.4767	0.0862
SVMGuide2	0.0717	0.0016	0.0462	0.0022	0.0461	0.0035
USPS	1.1786	0.1585	1.387	0.1657	2.2268	0.1774

5 Experimentos de Regressão com Outliers

Neste capítulo avalia-se o GOR-ELM. 13 bancos de dados para *multi-target regression* (MTR) e seis bancos de dados de regressão de uma saída são utilizados nos experimentos. Os dados foram obtidos da biblioteca do mulan (DŽEROSKI; DEMŠAR; GRBOVIĆ, 2000; SPYROMITROS-XIOUFIS et al., 2016) e do UCI (ASUNCION; NEWMAN, 2007). A Tabela 11 e Tabela 12 resumem os bancos de dados.

Tabela 11 – Informações sobre o banco de dados de MTR.

Banco	# Dados de treino	# Dados de teste	# Atributos	# Saídas
andro	25	24	30	6
atp1d	169	168	411	6
atp7d	148	148	411	6
enb	384	384	8	2
jura	180	179	15	3
oes10	202	201	298	16
oes87	167	167	263	16
rf1	4503	4502	64	8
rf2	3840	3839	576	8
scm1d	4902	4901	280	16
scm20d	4483	4483	61	16
slump	52	51	7	3
wq	530	530	16	14

Tabela 12 – Informações sobre o banco de dados de regressão de uma saída.

Banco	# Dados de treino	# Dados de teste	# Atributos
autoprice	80	79	15
balloon	1001	1000	1
housing	253	253	13
pyrim	37	37	27
space ga	1554	1553	6
strikes	313	312	6

No intuito de verificar a robustez à *outliers*, os bancos de dados de MTR são contaminados com *outliers* da seguinte forma:

1. Escolhe-se aleatoriamente, do conjunto de treino, uma amostra para ser contaminada.
2. Para cada linha da matriz de saídas da amostra obtida no item 1, todos os elementos dessa linha são substituídos por valores aleatórios de uma distribuição uniforme com

valores entre $Q_1 - 3(Q_3 - Q_1)$ e $Q_1 - 1.5(Q_3 - Q_1)$, onde Q_1 e Q_3 representam vetores do primeiro e terceiro quartil, respectivamente, da matriz de saída do conjunto de treino. Cada elemento do vetor de Q_1 e Q_3 representa o primeiro e terceiro quartil, respectivamente, de uma coluna da matriz de saídas do conjunto de treino.

3. Os passos 1 e 2 são repetidos, mas os dados da amostra selecionada são substituídos por valores aleatórios de uma distribuição uniforme com valores entre $Q_3 + 1.5(Q_3 - Q_1)$ e $Q_3 + 3(Q_3 - Q_1)$.

Essa metodologia segue como base a construção do gráfico do tipo boxplot, e os intervalos escolhidos para a distribuição uniforme foram estabelecidos de forma a extrapolar o limite superior e inferior do boxplot, sendo então considerados valores discrepantes.

Para os bancos de dados de regressão de uma saída, o mesmo procedimento foi adotado, contudo, Q_1 e Q_3 não são mais vetores, já que se tem apenas uma saída nesses bancos de dados.

Para os experimentos desse capítulo, 10%, 20%, 30% e 40% das saídas do conjunto de treino são selecionadas para serem contaminadas com *outliers*. Além disso, os experimentos são realizados sem a contaminação de valores discrepantes para possibilitar a comparação da robustez dos métodos testados.

Os métodos selecionados para comparação nos experimentos de robustez à *outliers* são: GOR-ELM, GR-ELM, R-ELM (com regularização *ridge* apenas) e o OR-ELM. Note que apenas o GOR-ELM e o OR-ELM possuem em sua construção abordagens que contemplem *outliers*. Portanto, espera-se um desempenho pior no GR-ELM e R-ELM (*ridge*) quando comparados ao GOR-ELM e OR-ELM, nos casos onde há presença de *outliers*.

Os experimentos de regressão desse capítulo foram conduzidos no MATLAB 8.5, em um computador com 3.6 GHz core i7 e 8 GB de RAM.

5.1 Especificação dos Parâmetros

O número inicial de neurônios, \tilde{N} , na camada oculta é de 1000 para todos os bancos de dados de regressão. Para o ADMM, fixa-se o $\rho = 1$ e os parâmetros de regularização λ_1 , C e τ são selecionados por validação cruzada com 5-fold. 31 valores para τ : $[2^{-10}, 2^{-9}, \dots, 2^{19}, 2^{20}]$ são testados. Para C , 14 valores foram testados: $[0.01, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000]$. Para λ_1 , 100 valores decrescentes foram testados variando-os de 100 a λ_{\min} em escala logarítmica. O valor de λ_{\min} é definido como 0.0001 se $N > n$ e 0.01 caso contrário. O parâmetro λ_2 é fixado em 0.1 para os bancos de dados MTR e 0.5 para os bancos de dados de regressão de uma saída. Os valores

de \mathbf{A} e $\boldsymbol{\nu}$ são gerado aleatoriamente baseados em uma distribuição uniforme e a função $h(\mathbf{x}_k; \mathbf{a}_i, \nu_i) = 1/(1 + \exp(-\mathbf{a}_i \cdot \mathbf{x} - \nu_i))$ é escolhida como função de ativação em todos os métodos. Para o GOR-ELM utiliza-se como critério de parada 30 iterações. Para o OR-ELM utiliza-se a recomendação de Zhang e Luo (2015), e termina-se o algoritmo com 20 iterações.

5.2 Avaliação do GOR-ELM

Nesta seção, os algoritmos GOR-ELM, GR-ELM, R-ELM e OR-ELM são comparados. Os bancos de dados são divididos em dois conjuntos de tamanhos aproximadamente iguais: um de treino e outro de teste. O conjunto de treino é usado para obter os parâmetros e os pesos de saída e o conjunto de teste é usado para avaliar a rede treinada em dados que não foram usados na etapa de treinamento.

De forma similar aos testes realizados no Capítulo 4, um experimento é definido da seguinte forma: novos valores de \mathbf{A} e $\boldsymbol{\nu}$ são gerados; realiza-se uma permutação aleatória dentro do conjunto de treino e teste; a rede neural é treinada usando o GOR-ELM, GR-ELM, R-ELM e o OR-ELM; e as métricas de avaliação são obtidas. Para cada banco de dados, 20 execuções do experimento são realizadas e a média das métricas são obtidas. As Tabelas 13, 14, 15, 16 e 17 resumem o erro médio de treino e teste (e os respectivos desvios padrões) para os bancos de dados de MTR.

Para o caso onde não há contaminação no banco de treino com *outliers* (Tabela 13) observa-se que o GR-ELM e R-ELM apresentam um valor de aRRMSE (definida na Equação 1.2) menor quando comparados com o GOR-ELM e OR-ELM. Esse comportamento já era esperado pois a minimização da norma Frobenius (e a norma ℓ_2) do erro é mais adequada para os casos onde não há *outliers*.

As Tabelas 14, 15, 16 e 17 resumem os resultados obtidos com contaminação de 10%, 20%, 30% e 40%, respectivamente, das saídas do conjunto de treino com *outlier*. Para esses casos, os métodos GOR-ELM e OR-ELM apresentaram melhor resultados quando comparados com o GR-ELM e R-ELM. Além disso, ao comparar o aRRMSE de teste do GOR-ELM e OR-ELM, tem-se que o GOR-ELM apresentou, no geral, um erro menor quando comparado ao OR-ELM. Uma possível explicação para isso é a metodologia adotada para contaminar o conjunto de treino. A contaminação em todos os elementos da linha da matriz de saída resulta em uma estrutura nos *outliers*. A penalização $\ell_{2,1}$ no erro consegue capturar tal estrutura e resulta em uma robustez maior à *outliers* com essa característica. Note ainda que a medida que a proporção de *outliers* aumenta, o OR-ELM apresenta resultados mais próximos aos obtidos pelo GOR-ELM.

As Tabelas 18, 19, 20, 21 e 22 exibem os resultados obtidos para os problemas de regressão com apenas uma saída. O intuito desses testes é verificar como o GOR-ELM se

comporta no caso onde há apenas uma saída.

De forma similar à observada nos testes com os conjuntos de MTR, tem-se que para o caso *em que* não há contaminação com *outliers* (Tabela 18) no banco de dados de treino, os métodos R-ELM e GR-ELM apresentaram um RRMSE menor quando comparados ao GOR-ELM e OR-ELM. Em especial, o R-ELM obteve, consistentemente, um RRMSE menor ou igual aos demais métodos testados.

As Tabelas 19, 20, 21 e 22 resumem os resultados obtidos para 10%, 20%, 30% e 40%, respectivamente, das amostras de conjunto de treino com *outliers*. Novamente, um resultado similar ao caso MTR é observado para o caso onde há apenas uma saída. Assim, nota-se que o GOR-ELM se mostra robusto à *outliers* tanto para os problemas de MTR quanto para problemas de regressão com apenas uma saída.

Tabela 13 – aRRMSE de Treino e Teste para os bancos de dados MTR ($\lambda_2 = 0.1$, $\widetilde{N} = 1000$ e sem *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste
andro	0.44 ± 0.04	0.92 ± 0.03	0.13 ± 0.00	0.69 ± 0.02	0.19 ± 0.00	0.68 ± 0.02	0.13 ± 0.00	0.69 ± 0.02
atp1d	0.42 ± 0.00	0.51 ± 0.01	0.15 ± 0.00	0.49 ± 0.01	0.14 ± 0.00	0.49 ± 0.01	0.35 ± 0.00	0.49 ± 0.01
atp7d	0.37 ± 0.01	0.68 ± 0.02	2.00 ± 0.09	1.66 ± 0.07	0.29 ± 0.00	0.68 ± 0.01	0.68 ± 0.01	0.81 ± 0.01
enb	0.32 ± 0.00	0.33 ± 0.00	0.12 ± 0.00	0.20 ± 0.00	0.19 ± 0.00	0.25 ± 0.00	0.06 ± 0.00	0.16 ± 0.01
jura	0.78 ± 0.03	0.95 ± 0.03	0.39 ± 0.00	0.57 ± 0.00	0.38 ± 0.00	0.57 ± 0.00	0.52 ± 0.00	0.61 ± 0.00
oes10	0.54 ± 0.01	0.75 ± 0.01	0.55 ± 0.00	0.73 ± 0.00	0.54 ± 0.00	0.73 ± 0.00	0.67 ± 0.00	0.79 ± 0.00
oes97	0.56 ± 0.01	0.77 ± 0.00	0.55 ± 0.00	0.75 ± 0.00	0.55 ± 0.00	0.75 ± 0.00	0.62 ± 0.00	0.79 ± 0.00
rf1	0.18 ± 0.00	0.14 ± 0.01	0.34 ± 0.02	0.36 ± 0.02	0.18 ± 0.00	0.17 ± 0.00	0.15 ± 0.00	0.11 ± 0.01
rf2	0.23 ± 0.01	0.27 ± 0.02	0.26 ± 0.00	0.33 ± 0.01	0.25 ± 0.00	0.32 ± 0.01	0.25 ± 0.01	0.31 ± 0.01
scm1d	0.35 ± 0.00	0.40 ± 0.00	0.33 ± 0.00	0.39 ± 0.00	0.33 ± 0.00	0.39 ± 0.00	0.36 ± 0.00	0.40 ± 0.00
scm20d	0.43 ± 0.00	0.50 ± 0.00	0.40 ± 0.00	0.49 ± 0.00	0.41 ± 0.00	0.49 ± 0.00	0.46 ± 0.00	0.52 ± 0.00
slump	0.51 ± 0.01	0.67 ± 0.01	0.46 ± 0.00	0.64 ± 0.00	0.46 ± 0.00	0.64 ± 0.00	0.66 ± 0.01	0.74 ± 0.01
wq	0.85 ± 0.00	0.92 ± 0.00	0.84 ± 0.00	0.92 ± 0.00	0.84 ± 0.00	0.92 ± 0.00	0.91 ± 0.00	0.98 ± 0.00

Tabela 14 – aRRMSE Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e 10% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste
andro	0.46 ± 0.15	1.21 ± 0.16	0.20 ± 0.05	1.29 ± 0.23	0.29 ± 0.06	1.20 ± 0.21	0.41 ± 0.09	1.07 ± 0.19
atp1d	0.63 ± 0.04	0.54 ± 0.02	0.29 ± 0.03	0.63 ± 0.05	0.28 ± 0.03	0.63 ± 0.05	0.63 ± 0.04	0.50 ± 0.01
atp7d	0.58 ± 0.04	0.69 ± 0.02	1.39 ± 0.10	1.47 ± 0.08	0.40 ± 0.03	0.76 ± 0.03	0.79 ± 0.03	0.81 ± 0.02
enb	0.87 ± 0.02	0.37 ± 0.03	0.65 ± 0.03	1.02 ± 0.14	0.75 ± 0.03	0.72 ± 0.10	0.82 ± 0.03	0.35 ± 0.16
jura	0.90 ± 0.06	0.96 ± 0.05	0.63 ± 0.04	0.76 ± 0.05	0.63 ± 0.04	0.76 ± 0.05	0.81 ± 0.04	0.62 ± 0.01
oes10	0.63 ± 0.03	0.76 ± 0.01	0.65 ± 0.03	0.75 ± 0.01	0.64 ± 0.03	0.75 ± 0.01	0.75 ± 0.03	0.80 ± 0.01
oes97	0.67 ± 0.03	0.78 ± 0.01	0.66 ± 0.03	0.77 ± 0.01	0.65 ± 0.03	0.77 ± 0.01	0.71 ± 0.03	0.80 ± 0.01
rf1	0.77 ± 0.02	0.14 ± 0.01	0.73 ± 0.01	0.39 ± 0.02	0.74 ± 0.02	0.31 ± 0.02	0.77 ± 0.02	0.16 ± 0.02
rf2	0.77 ± 0.02	0.35 ± 0.03	0.70 ± 0.02	0.62 ± 0.04	0.70 ± 0.02	0.60 ± 0.04	0.77 ± 0.02	0.41 ± 0.02
scm1d	0.85 ± 0.00	0.40 ± 0.00	0.79 ± 0.00	0.56 ± 0.01	0.79 ± 0.00	0.56 ± 0.01	0.85 ± 0.00	0.41 ± 0.00
scm20d	0.85 ± 0.01	0.51 ± 0.00	0.77 ± 0.00	0.72 ± 0.01	0.79 ± 0.00	0.66 ± 0.01	0.86 ± 0.01	0.52 ± 0.00
slump	0.81 ± 0.05	0.69 ± 0.05	0.68 ± 0.05	0.80 ± 0.08	0.68 ± 0.04	0.80 ± 0.08	0.88 ± 0.04	0.73 ± 0.04
wq	0.91 ± 0.01	0.93 ± 0.00	0.87 ± 0.01	0.96 ± 0.01	0.88 ± 0.01	0.95 ± 0.01	0.94 ± 0.01	0.99 ± 0.00

Tabela 15 – aRRMSE de Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e 20% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste
andro	0.45 ± 0.11	1.35 ± 0.27	0.21 ± 0.04	1.53 ± 0.28	0.31 ± 0.05	1.40 ± 0.26	0.45 ± 0.08	1.24 ± 0.23
atp1d	0.72 ± 0.04	0.59 ± 0.04	0.33 ± 0.03	0.77 ± 0.06	0.33 ± 0.03	0.77 ± 0.06	0.73 ± 0.05	0.52 ± 0.01
atp7d	0.63 ± 0.03	0.73 ± 0.03	1.13 ± 0.11	1.51 ± 0.09	0.43 ± 0.02	0.86 ± 0.05	0.82 ± 0.03	0.81 ± 0.03
enb	0.94 ± 0.01	0.47 ± 0.06	0.72 ± 0.03	1.46 ± 0.12	0.82 ± 0.02	1.02 ± 0.09	0.87 ± 0.03	0.63 ± 0.18
jura	0.93 ± 0.04	0.97 ± 0.04	0.70 ± 0.03	0.88 ± 0.07	0.70 ± 0.03	0.88 ± 0.07	0.88 ± 0.02	0.64 ± 0.02
oes10	0.69 ± 0.03	0.78 ± 0.02	0.72 ± 0.03	0.78 ± 0.02	0.72 ± 0.03	0.78 ± 0.02	0.81 ± 0.02	0.80 ± 0.01
oes97	0.73 ± 0.02	0.78 ± 0.01	0.71 ± 0.02	0.78 ± 0.02	0.71 ± 0.02	0.78 ± 0.02	0.76 ± 0.02	0.80 ± 0.01
rf1	0.89 ± 0.01	0.14 ± 0.00	0.83 ± 0.01	0.52 ± 0.03	0.84 ± 0.01	0.44 ± 0.03	0.88 ± 0.01	0.20 ± 0.01
rf2	0.87 ± 0.01	0.44 ± 0.04	0.79 ± 0.01	0.82 ± 0.03	0.79 ± 0.01	0.78 ± 0.03	0.87 ± 0.02	0.55 ± 0.04
scm1d	0.92 ± 0.01	0.41 ± 0.00	0.85 ± 0.00	0.70 ± 0.02	0.85 ± 0.00	0.71 ± 0.02	0.93 ± 0.01	0.41 ± 0.00
scm20d	0.92 ± 0.01	0.52 ± 0.00	0.84 ± 0.01	0.91 ± 0.02	0.85 ± 0.01	0.81 ± 0.02	0.93 ± 0.01	0.53 ± 0.00
slump	0.89 ± 0.05	0.73 ± 0.07	0.75 ± 0.05	0.89 ± 0.10	0.75 ± 0.05	0.89 ± 0.10	0.93 ± 0.04	0.77 ± 0.05
wq	0.93 ± 0.00	0.94 ± 0.00	0.89 ± 0.01	0.99 ± 0.01	0.90 ± 0.01	0.98 ± 0.01	0.94 ± 0.01	0.99 ± 0.01

Tabela 16 – aRRMSE de Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\widetilde{N} = 1000$ e 30% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste
andro	0.47 ± 0.11	1.67 ± 0.40	0.22 ± 0.04	1.87 ± 0.33	0.32 ± 0.06	1.68 ± 0.29	0.49 ± 0.08	1.49 ± 0.31
atp1d	0.79 ± 0.04	0.67 ± 0.07	0.38 ± 0.03	0.90 ± 0.08	0.37 ± 0.03	0.90 ± 0.08	0.82 ± 0.04	0.55 ± 0.03
atp7d	0.68 ± 0.04	0.79 ± 0.05	1.00 ± 0.09	1.55 ± 0.10	0.46 ± 0.03	0.97 ± 0.07	0.85 ± 0.02	0.84 ± 0.04
enb	0.96 ± 0.01	0.55 ± 0.09	0.72 ± 0.02	1.81 ± 0.16	0.83 ± 0.02	1.25 ± 0.11	0.85 ± 0.03	1.19 ± 0.21
jura	0.93 ± 0.04	1.01 ± 0.07	0.72 ± 0.02	1.02 ± 0.09	0.72 ± 0.02	1.02 ± 0.09	0.90 ± 0.02	0.66 ± 0.02
oes10	0.72 ± 0.03	0.79 ± 0.02	0.76 ± 0.03	0.79 ± 0.03	0.75 ± 0.03	0.79 ± 0.03	0.84 ± 0.03	0.81 ± 0.02
oes97	0.78 ± 0.02	0.80 ± 0.01	0.76 ± 0.02	0.80 ± 0.02	0.76 ± 0.02	0.80 ± 0.02	0.80 ± 0.02	0.81 ± 0.01
rf1	0.94 ± 0.01	0.15 ± 0.01	0.87 ± 0.01	0.62 ± 0.03	0.89 ± 0.01	0.53 ± 0.02	0.93 ± 0.01	0.24 ± 0.03
rf2	0.91 ± 0.01	0.67 ± 0.04	0.83 ± 0.01	0.98 ± 0.03	0.83 ± 0.01	0.93 ± 0.03	0.90 ± 0.01	0.76 ± 0.03
scm1d	0.95 ± 0.00	0.42 ± 0.00	0.88 ± 0.00	0.82 ± 0.02	0.88 ± 0.00	0.82 ± 0.02	0.96 ± 0.00	0.42 ± 0.00
scm20d	0.94 ± 0.00	0.54 ± 0.01	0.86 ± 0.00	1.05 ± 0.03	0.87 ± 0.00	0.93 ± 0.03	0.95 ± 0.00	0.54 ± 0.01
slump	0.90 ± 0.03	0.74 ± 0.07	0.76 ± 0.04	0.98 ± 0.11	0.76 ± 0.04	0.98 ± 0.11	0.94 ± 0.03	0.77 ± 0.05
wq	0.94 ± 0.01	0.94 ± 0.01	0.89 ± 0.01	1.02 ± 0.02	0.91 ± 0.01	1.00 ± 0.02	0.94 ± 0.01	1.00 ± 0.01

Tabela 17 – aRRMSE de Treino e Teste para os bancos de dados de MTR ($\lambda_2 = 0.1$, $\tilde{N} = 1000$ e 40% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste	aRRMSE Treino	aRRMSE Teste
andro	0.53 ± 0.10	1.88 ± 0.40	0.23 ± 0.04	2.09 ± 0.32	0.34 ± 0.05	1.87 ± 0.32	0.53 ± 0.07	1.68 ± 0.35
atp1d	0.82 ± 0.03	0.71 ± 0.07	0.38 ± 0.03	0.99 ± 0.10	0.38 ± 0.03	0.99 ± 0.10	0.85 ± 0.03	0.57 ± 0.03
atp7d	0.68 ± 0.03	0.85 ± 0.08	0.86 ± 0.07	1.60 ± 0.08	0.46 ± 0.02	1.03 ± 0.09	0.86 ± 0.02	0.84 ± 0.05
enb	0.97 ± 0.01	0.58 ± 0.10	0.74 ± 0.02	2.08 ± 0.14	0.84 ± 0.01	1.42 ± 0.15	0.83 ± 0.02	1.76 ± 0.24
jura	0.93 ± 0.03	1.05 ± 0.06	0.74 ± 0.03	1.11 ± 0.10	0.74 ± 0.03	1.12 ± 0.10	0.92 ± 0.02	0.70 ± 0.03
oes10	0.76 ± 0.03	0.81 ± 0.03	0.80 ± 0.03	0.83 ± 0.04	0.79 ± 0.03	0.83 ± 0.04	0.87 ± 0.03	0.82 ± 0.01
oes97	0.81 ± 0.02	0.81 ± 0.02	0.80 ± 0.02	0.83 ± 0.03	0.79 ± 0.02	0.83 ± 0.03	0.83 ± 0.02	0.83 ± 0.02
rf1	0.97 ± 0.01	0.16 ± 0.01	0.89 ± 0.01	0.74 ± 0.04	0.91 ± 0.01	0.63 ± 0.03	0.95 ± 0.01	0.34 ± 0.04
rf2	0.92 ± 0.01	0.96 ± 0.05	0.85 ± 0.01	1.10 ± 0.04	0.85 ± 0.01	1.04 ± 0.04	0.92 ± 0.01	1.01 ± 0.04
scm1d	0.97 ± 0.00	0.43 ± 0.00	0.90 ± 0.00	0.94 ± 0.02	0.90 ± 0.00	0.95 ± 0.02	0.97 ± 0.00	0.43 ± 0.00
scm20d	0.95 ± 0.00	0.57 ± 0.01	0.87 ± 0.00	1.19 ± 0.02	0.89 ± 0.00	1.04 ± 0.02	0.96 ± 0.00	0.56 ± 0.00
slump	0.93 ± 0.02	0.81 ± 0.12	0.78 ± 0.03	1.09 ± 0.17	0.78 ± 0.03	1.09 ± 0.17	0.96 ± 0.02	0.81 ± 0.07
wq	0.95 ± 0.00	0.96 ± 0.01	0.90 ± 0.01	1.06 ± 0.03	0.91 ± 0.01	1.03 ± 0.02	0.94 ± 0.01	1.02 ± 0.02

Tabela 18 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e sem *outlier*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste
autoprice	0.51 ± 0.00	0.68 ± 0.00	0.25 ± 0.00	0.58 ± 0.01	0.23 ± 0.00	0.58 ± 0.01	0.64 ± 0.01	0.76 ± 0.01
balloon	0.50 ± 0.00	0.51 ± 0.00	0.50 ± 0.00	0.51 ± 0.00	0.50 ± 0.00	0.51 ± 0.00	0.87 ± 0.00	0.87 ± 0.00
housing	0.71 ± 0.04	0.80 ± 0.04	1.04 ± 0.05	1.10 ± 0.05	0.23 ± 0.00	0.37 ± 0.01	0.71 ± 0.01	0.69 ± 0.01
pyrim	0.31 ± 0.01	0.77 ± 0.02	0.57 ± 0.05	0.93 ± 0.03	0.11 ± 0.00	0.83 ± 0.02	0.34 ± 0.01	0.78 ± 0.02
space ga	0.66 ± 0.00	0.64 ± 0.00	0.49 ± 0.00	0.53 ± 0.00	0.50 ± 0.00	0.53 ± 0.00	0.85 ± 0.00	0.84 ± 0.00
strikes	0.83 ± 0.00	0.89 ± 0.00	0.74 ± 0.02	0.92 ± 0.03	0.60 ± 0.00	0.75 ± 0.01	0.94 ± 0.00	0.98 ± 0.00

Tabela 19 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 10% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste
autoprice	0.81 ± 0.05	0.71 ± 0.05	0.59 ± 0.05	0.78 ± 0.12	0.57 ± 0.05	0.79 ± 0.13	0.87 ± 0.04	0.76 ± 0.01
balloon	0.63 ± 0.07	0.52 ± 0.03	0.63 ± 0.06	0.52 ± 0.02	0.63 ± 0.06	0.52 ± 0.02	0.90 ± 0.01	0.87 ± 0.01
housing	0.74 ± 0.05	0.80 ± 0.07	0.64 ± 0.03	0.94 ± 0.10	0.52 ± 0.04	0.57 ± 0.06	0.84 ± 0.03	0.71 ± 0.01
pyrim	0.72 ± 0.10	0.80 ± 0.04	0.48 ± 0.07	1.05 ± 0.16	0.28 ± 0.10	1.08 ± 0.16	0.73 ± 0.09	0.81 ± 0.04
space ga	0.84 ± 0.01	0.59 ± 0.01	0.81 ± 0.01	0.60 ± 0.01	0.82 ± 0.01	0.59 ± 0.01	0.94 ± 0.01	0.85 ± 0.01
strikes	0.87 ± 0.02	0.91 ± 0.04	0.76 ± 0.03	1.23 ± 0.12	0.79 ± 0.02	0.96 ± 0.07	0.98 ± 0.01	0.98 ± 0.00

Tabela 20 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 20% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste
autoprice	0.88 ± 0.04	0.75 ± 0.08	0.65 ± 0.03	0.95 ± 0.15	0.63 ± 0.03	0.98 ± 0.15	0.93 ± 0.03	0.78 ± 0.02
balloon	0.74 ± 0.07	0.55 ± 0.06	0.73 ± 0.07	0.56 ± 0.06	0.73 ± 0.07	0.56 ± 0.06	0.92 ± 0.01	0.89 ± 0.02
housing	0.78 ± 0.04	0.83 ± 0.08	0.60 ± 0.03	1.08 ± 0.11	0.63 ± 0.03	0.73 ± 0.07	0.91 ± 0.02	0.72 ± 0.02
pyrim	0.79 ± 0.08	0.82 ± 0.08	0.47 ± 0.05	1.20 ± 0.20	0.31 ± 0.06	1.32 ± 0.22	0.79 ± 0.07	0.83 ± 0.08
space ga	0.91 ± 0.01	0.62 ± 0.01	0.89 ± 0.01	0.66 ± 0.02	0.90 ± 0.01	0.64 ± 0.02	0.97 ± 0.00	0.87 ± 0.01
strikes	0.90 ± 0.01	1.01 ± 0.07	0.79 ± 0.02	1.49 ± 0.12	0.83 ± 0.02	1.13 ± 0.09	0.99 ± 0.01	0.98 ± 0.01

Tabela 21 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 30% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste
autoprice	0.92 ± 0.04	0.84 ± 0.14	0.70 ± 0.04	1.12 ± 0.25	0.68 ± 0.04	1.15 ± 0.25	0.97 ± 0.03	0.79 ± 0.04
balloon	0.82 ± 0.07	0.57 ± 0.08	0.81 ± 0.07	0.60 ± 0.06	0.81 ± 0.07	0.60 ± 0.06	0.94 ± 0.01	0.90 ± 0.02
housing	0.81 ± 0.03	0.90 ± 0.07	0.62 ± 0.03	1.21 ± 0.11	0.69 ± 0.03	0.86 ± 0.10	0.95 ± 0.02	0.74 ± 0.02
pyrim	0.83 ± 0.08	0.84 ± 0.08	0.47 ± 0.07	1.36 ± 0.27	0.33 ± 0.07	1.48 ± 0.28	0.84 ± 0.08	0.85 ± 0.08
space ga	0.95 ± 0.01	0.67 ± 0.03	0.92 ± 0.01	0.73 ± 0.03	0.93 ± 0.01	0.70 ± 0.03	0.99 ± 0.00	0.88 ± 0.01
strikes	0.90 ± 0.01	1.10 ± 0.06	0.78 ± 0.02	1.78 ± 0.19	0.84 ± 0.01	1.31 ± 0.10	1.00 ± 0.01	0.98 ± 0.01

Tabela 22 – RRMSE de Treino e Teste para os bancos de dados de uma saída ($\lambda_2 = 0.5$, $\tilde{N} = 1000$ e 40% do conjunto de treino com *outliers*).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste	RRMSE Treino	RRMSE Teste
autoprice	0.93 ± 0.03	0.86 ± 0.14	0.71 ± 0.04	1.29 ± 0.23	0.69 ± 0.04	1.34 ± 0.23	0.98 ± 0.02	0.82 ± 0.06
balloon	0.88 ± 0.03	0.63 ± 0.09	0.86 ± 0.04	0.64 ± 0.05	0.86 ± 0.04	0.64 ± 0.05	0.95 ± 0.02	0.90 ± 0.03
housing	0.81 ± 0.03	0.98 ± 0.08	0.62 ± 0.03	1.41 ± 0.12	0.71 ± 0.03	0.99 ± 0.10	0.97 ± 0.01	0.76 ± 0.02
pyrim	0.83 ± 0.05	0.91 ± 0.13	0.47 ± 0.06	1.48 ± 0.23	0.33 ± 0.07	1.66 ± 0.25	0.84 ± 0.05	0.92 ± 0.12
space ga	0.96 ± 0.01	0.73 ± 0.03	0.94 ± 0.01	0.81 ± 0.04	0.94 ± 0.01	0.78 ± 0.04	1.00 ± 0.00	0.91 ± 0.01
strikes	0.91 ± 0.02	1.19 ± 0.09	0.79 ± 0.02	1.95 ± 0.17	0.84 ± 0.02	1.44 ± 0.11	1.00 ± 0.01	0.98 ± 0.01

Os testes estatísticos de Friedman e de Nemenyi foram realizados para verificar a ocorrência de diferença estatística no aRRMSE de treino e teste. Os testes estatísticos foram aplicados nos dados de MTR e regressão com uma saída, conjuntamente. Para o teste de Friedman, rejeitou-se a hipótese nula em todos os casos, isto é, os métodos não são equivalentes. A Figura 4 exibe a comparação entre os métodos, com relação ao aRRMSE, usando o teste de Nemenyi. Note que os métodos conectados entre si por uma linha espessa não apresentam diferença estatística. A diferença crítica (DC) representa a diferença necessária entre os *ranks* de dois métodos para serem considerados estatisticamente diferentes.

Ao analisar a Figura 4 observa-se que para o treino e teste, no caso onde não há *outliers* o R-ELM foi estatisticamente melhor que o GOR-ELM e OR-ELM. No caso de treino com *outliers* (qualquer proporção), tem-se que o R-ELM e o GR-ELM se mostraram estatisticamente melhores que o GOR-ELM. Para os testes com 10%, 20% e 30% de *outliers*, observa-se que o GOR-ELM foi estatisticamente melhor que o R-ELM e o GR-ELM, ao passo que não houve diferença estatística entre os demais métodos. Para o caso de teste com 40% de *outliers*, além do GOR-ELM, o OR-ELM também se mostrou estatisticamente melhor que o R-ELM e o GR-ELM.

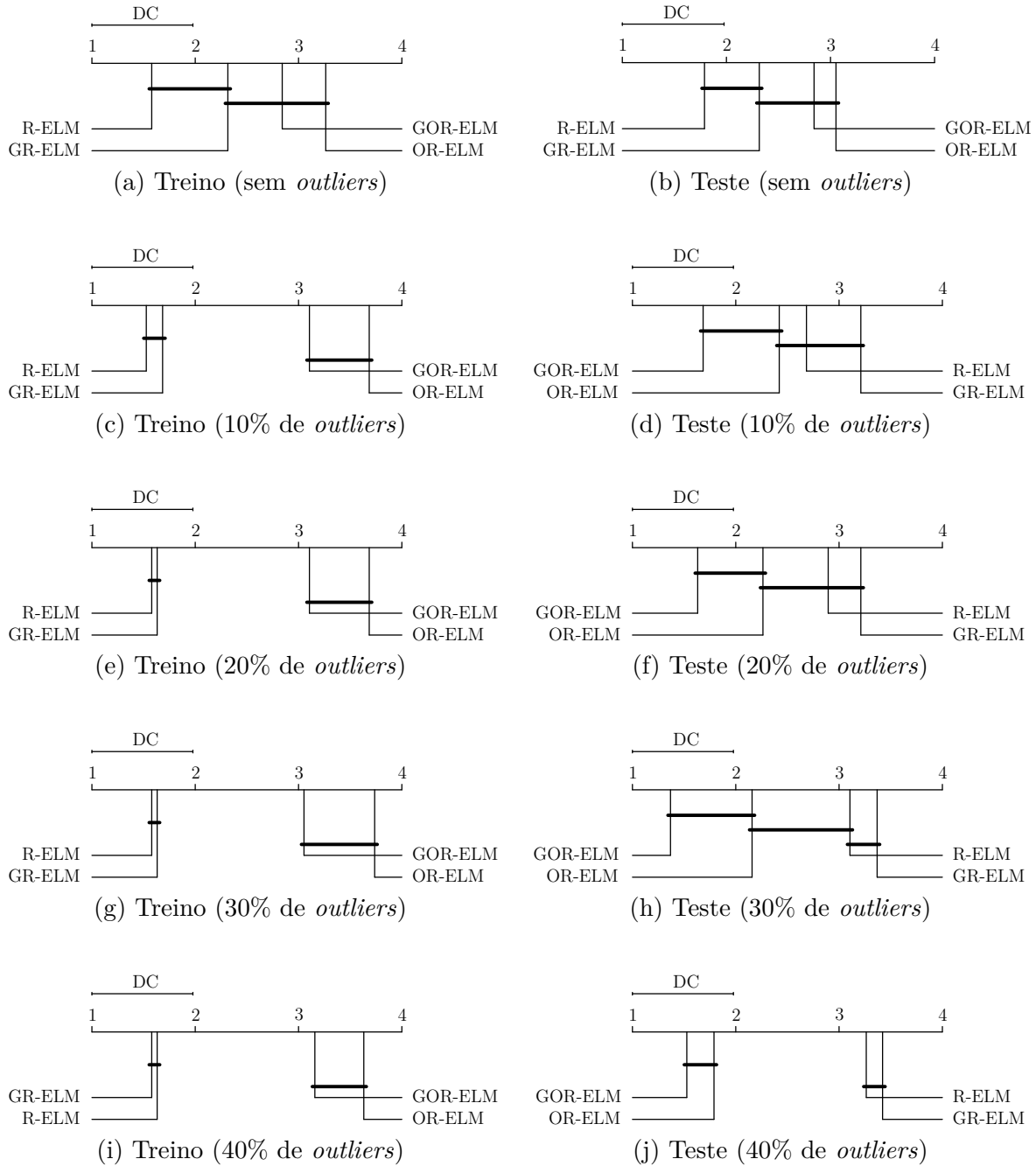
As Tabelas 23 e 24 resumem os parâmetros obtidos no treino pela validação cruzada com 5-*fold* para os bancos de dados de MTR e regressão com apenas uma saída, respectivamente.

Tabela 23 – Especificações dos parâmetros para os bancos de dados de MTR ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GOR-ELM		GR-ELM		R-ELM	OR-ELM
	λ_1	τ	λ_1	C	C	τ
andro	0.6579	256	0.0100	20	100	1
atp1d	0.0100	256	1.6681	2	20	1
atp7d	10.7227	256	57.2237	5	10	2
enb	0.0001	512	0.0001	1000	1000	0.0010
jura	3.0539	128	0.0614	5	50	8
oes10	2.0092	2048	3.1993	0.10	1	0.5000
oes97	29.8365	1024	0.0100	0.10	1	0.2500
rf1	3.5112	512	49.7702	50	100	0.0078
rf2	37.6494	1024	24.7708	5	20	0.1250
scm1d	57.2237	512	9.3260	0.50	5	0.5000
scm20d	57.2237	2048	100.00	5	20	0.1250
slump	0.1417	64	0.0001	5	50	8
wq	0.2477	8	0.2477	2	10	8

A Tabela 25 apresenta o número médio de neurônios obtidos nos métodos GOR-ELM e GR-ELM para os problemas de MTR com as diferentes proporções de contaminação

Figura 4 – Comparação entre os métodos em relação ao aRRMSE de treino e teste usando o teste de Nemenyi com diferentes proporções de *outliers* no banco de dados de treino.



Fonte: Produção do próprio autor.

da saída do conjunto de treino. Os problemas de regressão, em geral, precisam de um número maior de neurônios. Assim, em poucas bases de dados houve uma redução no número de neurônios. Essa constatação está alinhada com a expectativa para o método GOR-ELM e GR-ELM, já que se busca uma rede mais compacta mas mantendo um erro tão pequeno quanto ao obtido pelo OR-ELM e R-ELM. Se ao deixar a rede mais compacta,

Tabela 24 – Especificações dos parâmetros para os bancos de dados de MTR ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GOR-ELM		GR-ELM		R-ELM	OR-ELM
	λ_1	τ	λ_1	C	C	τ
autoprice	0.0001	131072	0.0001	20	50	0.1250
balloon	0.0001	16	0.0038	1000	1000	8192
housing	3.5112	1024	12.3285	500	200	128
pyrim	0.0001	0.25	0.0057	20	100	512
space ga	0.0001	64	0.0001	1000	1000	65536
strikes	0.0007	64	1.0000	1000	500	2048

o erro é muito penalizado tornando-o grande, então, é melhor que a rede permaneça com o número inicial de neurônios.

As mesmas observações podem ser feitas para os problemas de regressão com apenas uma saída (Tabela 26). Vale ressaltar que para os métodos R-ELM e OR-ELM o número inicial e final de neurônios é 1000, já que esses métodos não objetivam uma rede mais compacta.

Por fim, as Tabelas 27 e 28 exibem a média dos tempos médios das diferentes proporções de *outliers* para os conjuntos de MTR e regressão com apenas uma saída, respectivamente. A construção dessas tabelas segue da seguinte forma: após realizar 20 experimentos no conjunto de treino, obtêm-se para cada proporção de contaminação com *outliers* os registros dos tempos médios referentes às execuções desses 20 experimentos. Em seguida, obtém-se a média, desses tempos médios, referentes às diferentes proporções de *outliers*. Tal abordagem foi realizada para diminuir a quantidade de tabelas exibidas neste capítulo. Além disso, os valores dos tempos médios não variaram muito para as diferentes proporções de *outliers* e, portanto, a média desses tempos médios resumem bem os tempos de processamento de treino e teste para todos os métodos.

Note que o R-ELM possui um tempo de treino menor, quando comparado com os demais métodos. Isso já era esperado uma vez que o R-ELM precisa resolver apenas uma pseudo-inversa, ao passo que os demais métodos se baseiam em um algoritmo iterativo para solução do problema. Em compensação, o tempo de teste para todos os métodos foram similares. Os tempos de testes do GOR-ELM e GR-ELM não foram significativamente menores que os tempos do OR-ELM e R-ELM pois os problemas de regressão demandaram mais neurônios para suas soluções. Com isso, o tempo de teste não possui uma queda significativa, como a observada nos problemas de classificação.

Tabela 25 – Média do número de neurônios ao final do treinamento para cada banco de dados de MTR nos métodos GOR-ELM e GR-ELM.

Banco	0%		10%		20%		30%		40%	
	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM
andro	907.30	1000.00	926.25	1000.00	941.10	1000.00	952.55	1000.00	956.85	1000.00
atp1d	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00
atp7d	946.35	465.90	946.50	522.20	938.90	556.45	936.50	574.35	933.70	599.20
enb	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00
jura	539.95	999.00	537.80	999.75	540.85	999.65	549.35	999.75	562.65	999.85
oes10	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00
oes97	999.80	1000.00	1000.00	1000.00	999.95	1000.00	999.95	1000.00	999.85	1000.00
rf1	1000.00	832.65	1000.00	923.35	1000.00	952.65	1000.00	970.25	1000.00	975.90
rf2	999.60	991.10	999.55	995.20	999.30	996.30	999.55	997.30	999.25	997.50
scm1d	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00
scm20d	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00
slump	998.85	1000.00	998.65	1000.00	998.65	1000.00	999.15	1000.00	998.40	1000.00
wq	998.15	993.75	998.25	995.75	997.70	995.75	996.25	994.20	993.35	992.70

Tabela 26 – Média do número de neurônios ao final do treinamento para cada banco de dados de regressão de uma saída nos métodos GOR-ELM e GR-ELM.

Banco	0%		10%		20%		30%		40%	
	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM	GOR-ELM	GR-ELM
autoprice	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00
balloon	999.70	979.80	999.45	979.30	999.50	980.65	999.50	980.75	999.30	976.10
housing	644.50	512.00	692.85	668.45	728.65	736.35	747.15	773.15	776.60	792.15
pyrim	997.75	839.45	997.00	880.60	998.05	900.70	997.75	913.90	997.85	924.05
space ga	999.10	999.95	999.70	1000.00	999.80	999.95	999.80	999.95	999.80	1000.00
strikes	994.45	541.25	998.90	649.10	999.20	693.95	999.55	740.25	999.35	762.35

Tabela 27 – Média dos tempos médios de treino e teste para os bancos de dados de MTR ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	Treino	Teste	Treino	Teste	Treino	Teste	Treino	Teste
andro	0.0063	0.0005	0.0059	0.0005	0.0001	0.0005	0.0019	0.0005
atp1d	0.0017	0.0038	0.0158	0.0037	0.0010	0.0037	0.0137	0.0039
atp7d	0.0131	0.0034	0.0142	0.0020	0.0008	0.0035	0.0112	0.0035
enb	0.0049	0.0052	0.0247	0.0053	0.0037	0.0050	0.0423	0.0051
jura	0.0151	0.0014	0.0108	0.0022	0.0010	0.0022	0.0128	0.0023
oes10	0.0045	0.0041	0.0405	0.0040	0.0013	0.0041	0.0200	0.0043
oes97	0.0121	0.0034	0.0323	0.0034	0.0010	0.0034	0.0158	0.0035
rf1	0.3438	0.0835	0.2943	0.0823	0.0498	0.0834	0.5137	0.0830
rf2	0.4546	0.1097	0.2718	0.1022	0.0499	0.1059	0.5718	0.1047
scm1d	0.3567	0.1110	0.5196	0.1081	0.0599	0.1084	0.7197	0.1103
scm20d	0.3242	0.0841	0.4896	0.0840	0.0549	0.0835	0.6178	0.0844
slump	0.0027	0.0008	0.0045	0.0008	0.0002	0.0008	0.0027	0.0008
wq	0.0787	0.0073	0.1484	0.0072	0.0065	0.0071	0.0791	0.0070

Tabela 28 – Média dos tempos médios de treino e teste para os bancos de dados de regressão de uma saída ($\lambda_2 = 0.1$ e $\tilde{N} = 1000$).

Banco	GOR-ELM		GR-ELM		R-ELM		OR-ELM	
	Treino	Teste	Treino	Teste	Treino	Teste	Treino	Teste
autoprice	0.0008	0.0013	0.0033	0.0013	0.0003	0.0013	0.0032	0.0012
balloon	0.0902	0.0139	0.0424	0.0136	0.0232	0.0134	0.2315	0.0136
housing	0.0096	0.0024	0.0047	0.0023	0.0018	0.0032	0.0175	0.0034
pyrim	0.0021	0.0007	0.0009	0.0006	0.0001	0.0007	0.0016	0.0007
space ga	0.0957	0.0307	0.0541	0.0303	0.0276	0.0297	0.3279	0.0292
strikes	0.0117	0.0040	0.0064	0.0025	0.0025	0.0039	0.0253	0.0041

6 Conclusão

Na literatura de ELM, diversas variantes do ELM têm sido aplicadas com sucesso em várias áreas do conhecimento. Pelo fato dos pesos de entrada serem atribuídos aleatoriamente, os treinamentos de redes SLFN usando ELM e suas variantes são rápidos. Ademais, a literatura mostra que esses métodos possuem boa generalização, quando regularizados, e geralmente possuem desempenho melhor que o tradicional SVM quando a tarefa é classificação. Contudo, a maioria dos esforços nessa área tem sido para resolver problemas onde só há uma saída, e as extensões para os problemas de múltiplas saídas, em geral consistem em aplicar o método tradicional (para uma saída) em cada uma das saídas de forma independente, isto é, não levam em consideração qualquer tipo de estrutura nessas saídas. Além disso, pouco trabalho tem sido desenvolvido em ELM para tratar de problemas com *outliers*. Outro problema que recentemente ganhou força na área são os algoritmos para paralelizar os métodos baseados em ELM. Embora avanços tenham sido feitos, esses não contemplam a compacidade da rede nem problemas com *outliers*.

Nesse trabalho foi proposto uma generalização do ELM regularizados, para problemas de múltiplas saídas, em especial classificação multiclasse e MTR. Para o problema de classificação multiclasse, o GR-ELM foi proposto como generalização do R-ELM. Já para os problemas de MTR foi proposto o GOR-ELM como generalização do OR-ELM. Como característica dos métodos propostos têm-se: busca por uma rede mais compacta sem prejudicar a acurácia ou o erro do método; possui desempenho similar aos métodos tradicionais nos problemas de uma saída e não apenas nos problemas de múltiplas saídas; são facilmente adaptáveis para serem executados de forma distribuída; e naturalmente levam em conta a estrutura da saída, ao contrário dos demais métodos que tratam as mesmas de forma independente.

O R-ELM utiliza a regularização *elastic net* para, ao mesmo tempo que determina os pesos que conectam a camada oculta à camada de saída, também busca por uma rede mais compacta. Essa abordagem foi proposta para problemas de regressão de uma saída, mas sua extensão para problemas de classificação binária é natural. Para problemas de classificação multiclasse, uma alternativa seria usar o R-ELM em um esquema de *one-versus-rest*, que geralmente é evitada, já que quando há um número grande de classes o custo computacional se torna demasiadamente elevado. Além disso, na proposta do R-ELM não há preocupação em aplicações distribuídas. Nesse sentido, o GR-ELM e DGR-ELM são propostos como generalização do R-ELM na tentativa de superar esses problemas. Para atingir tal generalização, utiliza-se a norma Frobenius como extensão natural da norma ℓ_2 , e a norma $\ell_{2,1}$ como generalização da norma ℓ_1 . O algoritmo escolhido para resolver o problema de otimização resultante é o ADMM que, no caso GR-ELM, pode ser

facilmente adaptado para problemas distribuídos, originando o DGR-ELM.

O OR-ELM surge para tratar problemas com *outliers*, que têm crescido com o advento da era do *big data*. A estratégia do OR-ELM é usar a norma ℓ_1 , no lugar da norma ℓ_2 , como função de perda. Entretanto, o OR-ELM leva em consideração apenas a regularização *ridge* e, portanto, os pesos obtidos são densos. Além disso, como o OR-ELM faz uso do ALM para resolver o problema de otimização resultante, o mesmo não pode ser aplicado no contexto distribuído. Neste trabalho, o GOR-ELM foi proposto como tentativa de superar as limitações do OR-ELM, para problemas de MTR com *outliers*. Para isso, utilizou-se o GR-ELM e a norma Frobenius do erro foi substituída pela norma $\ell_{2,1}$. No caso particular, quando há apenas uma saída e quando o parâmetro de regularização, que leva em conta a compacidade da rede, é zero, obtém-se o OR-ELM. Como o algoritmo usado para resolver o GOR-ELM é o ADMM, segue que o método é facilmente adaptável para problemas distribuídos, embora não tenha sido desenvolvido nesta Tese.

Para avaliar os métodos propostos nesta Tese, 41 bases de dados foram testadas. Dessas, 22 são de classificação, sendo 14 de classificação binária e oito de classificação multiclasse; e 19 são de regressão, sendo seis de regressão de uma saída e 13 de MTR. Nos problemas de regressão, *outliers* foram inseridos com base na construção do boxplot. Taxas de 10%, 20%, 30% e 40% de *outliers* foram aleatoriamente inseridos no conjunto de treino para testar a robustez à *outliers* do GOR-ELM. Para determinar os parâmetros utilizados nesta Tese, foi realizado validação cruzada com 5-*fold*. Todos os experimentos iniciaram com 1000 neurônios. Além disso, 20 execuções de cada experimento foram realizadas a fim de se obter a média das métricas de avaliação.

Nos problemas de classificação, o GR-ELM foi comparado com o R-ELM com regularização *ridge*. Como métrica de avaliação, a acurácia de treino e teste foi obtida. Além disso, o tempo de treinamento e teste foi computado para cada base de dados. Os resultados indicam que o GR-ELM obteve resultados de acurácia similares ao R-ELM, tanto para os problemas de classificação binária quanto para multiclasse. Além disso, os resultados indicam que, em geral, o GR-ELM resulta em uma rede mais compacta, já que precisa de menos neurônios para obter uma acurácia similar ao R-ELM. O tempo de treinamento do GR-ELM é maior, entretanto, o tempo de teste tende a ser menor. Resultados similares foram obtidos para DGR-ELM. Nesse caso, foram testados o DGR-ELM com 1, 2 e 4 processadores.

Nos problemas de regressão, o GOR-ELM foi comparado com o OR-ELM, GR-ELM e R-ELM com regularização *ridge*. Nesse caso, o aRRMSE de treino e teste foi usado como métrica de avaliação. De forma similar à avaliação dos problemas de classificação, para os problemas de regressão foram mensurados o tempo de treino e teste. Os resultados indicam que quando não há presença de *outliers* os métodos GR-ELM e R-ELM apresentam desempenho melhor quando comparados ao OR-ELM e GOR-ELM. Entretanto, quando

a proporção de *outliers* aumenta, os métodos GOR-ELM e OR-ELM passam a ter um desempenho melhor se comparado ao R-ELM e GR-ELM. Os experimentos indicam que o número de neurônios necessários para resolver os problemas de regressão não sofreu muita alteração no GOR-ELM e nem no GR-ELM. Apesar desse ponto parecer um problema, na verdade essa é uma característica desejável nos métodos propostos. Isto é, busca-se por uma rede mais compacta sem afetar a acurácia ou erro. Assim, os problemas de regressão se mostraram mais custosos (do ponto de vista de quantidade de neurônios) e, portanto, poucos foram os casos onde houveram redução no número de neurônios.

Nesta Tese, foram propostas generalizações do ELM regularizados para problemas de múltiplas saídas, com regularizações que permitem obter uma rede mais compacta, e que possa ser implementada de forma distribuída, para tratar problemas com *outliers*. É possível destacar que:

- os métodos propostos são facilmente adaptáveis para problemas distribuídos;
- modificando-se a norma o GR-ELM foi possível tratar problemas de regressão com *outliers*; e
- a busca por uma rede mais compacta não afetou a acurácia ou erro dos métodos propostos.

Chega-se a conclusão de que a generalização do ELM regularizado, através de normas matriciais, contemplando a compacidade da rede, foi adequada para tratar problemas de classificação multiclasse e regressão de múltiplas saídas, que por sua vez podem estar contaminadas com *outliers*, e facilmente adaptável para problemas distribuídos.

Para trabalhos futuros pode-se focar nas seguintes observações:

- Embora o GOR-ELM tenha sido aplicado apenas para regressão, esse pode ser explorado para classificação;
- Nas simulações realizadas nesta Tese, apenas as SLFN com funções de ativação sigmoide foram testados. Extensões para outros tipos de funções de ativação, bem como a utilização de *kernel* podem ser realizadas;
- Melhoras no tempo de treinamento do DGR-ELM podem ser obtidas através do uso de pacotes de álgebra linear baseadas em GPU e uso do LAPACK ao invés do GLS para fatoração de Cholesky;
- Como critério de parada para o GOR-ELM foi usado um número máximo de iterações. Um critério baseado no erro dual e primal podem diminuir o tempo de treinamento do GOR-ELM;

- Embora não tenha sido realizado nesta Tese, o GOR-ELM, assim como o GR-ELM, é facilmente adaptável para tratar problemas distribuídos; e
- Apesar desta Tese ter explorado apenas os problemas de regressão e classificação, sem nenhuma mudança nos métodos, é possível aplica-los em problemas de classificação *multilabel*.

Referências

AHO, T.; ZENKO, B.; DZEROSKI, S.; ELOMAA, T. Multi-Target Regression with Rule Ensembles. *Journal of Machine Learning Research*, v. 13, p. 2367–2407, 2012. ISSN 1532-4435. Citado 3 vezes nas páginas 21, 22 e 25.

AMALDI, E.; KANN, V. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, v. 209, n. 1-2, p. 237–260, 1998. ISSN 03043975. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0304397597001151>>. Citado na página 28.

APPICE, A.; DZEROSKI, S. Stepwise induction of multi-target model trees. *Machine Learning: ECML 2007, Proceedings*, v. 4701, p. 502–509, 2007. Citado na página 22.

ASUNCION, A.; NEWMAN, D. J. *UCI Machine Learning Repository*. 2007. Disponível em: <www.ics.uci.edu/~mlearn/MLRepository.htm>. Citado 2 vezes nas páginas 49 e 59.

BAI, Z.; HUANG, G.-B.; WANG, D.; WANG, H.; WESTOVER, M. B. Sparse Extreme Learning Machine for Classification. *IEEE Transactions on Cybernetics*, v. 44, n. 10, p. 1858–1870, 2014. ISSN 2168-2267. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6756997>>. Citado 2 vezes nas páginas 20 e 51.

BOYD, S. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine Learning*, v. 3, n. 1, p. 1–122, 2010. ISSN 1935-8237. Disponível em: <<http://www.nowpublishers.com/article/Details/MAL-016>>. Citado 11 vezes nas páginas 30, 31, 33, 38, 40, 41, 42, 43, 47, 56 e 99.

BRANKE, J. Evolutionary algorithms for neural network design and training. In: *In Proceedings of the First Nordic Workshop on Genetic Algorithms and its Applications*. [S.l.: s.n.], 1995. Citado 2 vezes nas páginas 19 e 20.

BREIMAN, L.; FRIEDMAN, J. H. Predicting Multivariate Responses in Multiple Linear Regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, v. 59, n. 1, p. 3–54, feb 1997. ISSN 1369-7412. Disponível em: <<http://doi.wiley.com/10.1111/1467-9868.00054>>. Citado na página 22.

BROWN, P. J.; ZIDEK, J. V. Adaptive multivariate ridge regression. *The Annals of Statistics*, v. 8, n. 1, p. 64–74, 1980. ISSN 0090-5364. Disponível em: <<http://www.jstor.org/stable/2240743>>. Citado na página 22.

CHANG, C.-C.; LIN, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, v. 2, n. 3, p. 27:1—27:27, 2011. Citado na página 49.

CHEN, C.; HE, B.; YE, Y.; YUAN, X. The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. *Mathematical Programming, Springer*, v. 155, n. 1-2, p. 57–79, 2014. Citado na página 48.

CHEN, C.; LI, Y.; YAN, C.; GUO, J.; LIU, G. Least absolute deviation-based robust support vector regression. *Knowledge-Based Systems*, Elsevier B.V., v. 131, p. 183–194, 2017. ISSN 09507051. Disponível em: <<http://dx.doi.org/10.1016/j.knosys.2017.06.009>>. Citado 2 vezes nas páginas 28 e 44.

CHEN, C.-T. *Linear System Theory and Design*. 3. ed. [S.l.]: Oxford University Press, 1999. 46—47 p. Citado na página 109.

CHEN, K.; LV, Q.; LU, Y.; DOU, Y. Robust regularized extreme learning machine for regression using iteratively reweighted least squares. *Neurocomputing*, v. 230, n. June 2016, p. 345–358, mar 2017. ISSN 09252312. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0925231216315053>>. Citado na página 21.

CHEN, S.; COWAN, C. F. N.; GRANT, P. M. *Orthogonal least squares learning algorithm for radial basis function networks*. 1991. 302–309 p. Disponível em: <<http://www.ncbi.nlm.nih.gov/pubmed/18252625>>. Citado 2 vezes nas páginas 19 e 20.

CHEN, S. S.; DONOHO, D. L.; SAUNDERS, M. A. Atomic Decomposition by Basis Pursuit. *SIAM Journal on Scientific Computing*, v. 20, n. 1, p. 33–61, 1998. ISSN 1064-8275. Disponível em: <<http://epubs.siam.org/doi/10.1137/S1064827596304010>>. Citado na página 28.

CHOROWSKI, J.; WANG, J.; ZURADA, J. M. Review and performance comparison of SVM- and ELM-based classifiers. *Neurocomputing*, v. 128, p. 507–516, mar 2014. ISSN 09252312. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0925231213008825>>. Citado 2 vezes nas páginas 20 e 51.

DEMŠAR, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, v. 7, p. 1–30, 2006. ISSN 1532-4435. Citado 3 vezes nas páginas 111, 112 e 113.

DENG, W.; ZHENG, Q.; CHEN, L. Regularized Extreme Learning Machine. In: *2009 IEEE Symposium on Computational Intelligence and Data Mining*. [S.l.]: IEEE, 2009. p. 389–395. ISBN 978-1-4244-2765-9. Citado 3 vezes nas páginas 20, 34 e 35.

DONOHO, D. L.; JOHNSTONE, I.; MONTANARI, A. Accurate Prediction of Phase Transitions in Compressed Sensing via a Connection to Minimax Denoising. *IEEE Transactions on Information Theory*, v. 59, n. 6, p. 3396–3433, jun 2013. ISSN 0018-9448. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6409458>>. Citado na página 40.

DŽEROSKI, S.; DEMŠAR, D.; GRBOVIĆ, J. Predicting Chemical Parameters of River Water Quality from Bioindicator Data. *Applied Intelligence*, v. 13, n. 1, p. 7–17, 2000. ISSN 0924669X. Disponível em: <<http://link.springer.com/10.1023/A:1008323212047>>. Citado 2 vezes nas páginas 43 e 59.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, v. 33, n. 1, p. 1–22, 2010. ISSN 1548-7660. Disponível em: <<http://www.jstatsoft.org/v33/i01/>>. Citado na página 35.

FRIEDMAN, M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, v. 32, n. 200, p. 675–701, 1937. ISSN 1537274X. Citado na página 111.

GHOSN, J.; BENGIO, Y. Multi-Task Learning for Stock Selection. *Advances in Neural Information Processing Systems 9 (NIPS'96)*, n. January 1996, p. 946–952, 1997. Disponível em: <<http://www.iro.umontreal.ca/~lisa/pointeurs/multitask-nips97.p>>. Citado na página 43.

HAGAN, M.; MENHAJ, M. Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, v. 5, n. 6, p. 989–993, 1994. ISSN 10459227. Disponível em: <<http://ieeexplore.ieee.org/document/329697/>>. Citado 2 vezes nas páginas 19 e 20.

HAYKIN, S. Neural Networks: A Comprehensive Foundation. In: . [S.l.]: Prentice Hall, 1999. ISBN 9780132733502. Citado na página 19.

HODGE, V. J.; AUSTIN, J. A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review*, v. 22, n. 2, p. 85–126, 2004. ISSN 1573-7462. Disponível em: <<https://doi.org/10.1007/s10462-004-4304-y>>. Citado 2 vezes nas páginas 21 e 43.

HOERL, A. E.; KENNARD, R. W. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, v. 12, n. 1, p. 55–67, 1970. ISSN 0040-1706. Disponível em: <<http://www.tandfonline.com/doi/abs/10.1080/00401706.1970.10488634>>. Citado na página 35.

HORATA, P.; CHIEWCHANWATTANA, S.; SUNAT, K. Robust extreme learning machine. *Neurocomputing*, v. 102, n. Supplement C, p. 31–44, 2013. ISSN 0925-2312. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0925231212004171>>. Citado na página 21.

HUANG, G.; HUANG, G.-B.; SONG, S.; YOU, K. Trends in extreme learning machines: A review. *Neural Networks*, Elsevier Ltd, v. 61, p. 32–48, jan 2015. ISSN 08936080. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0893608014002214><http://linkinghub.elsevier.com/retrieve/pii/S0893608014002214>>. Citado 2 vezes nas páginas 19 e 20.

HUANG, G. B. Learning capability and storage capacity of two-hidden-layer feedforward networks. *IEEE Transactions on Neural Networks*, v. 14, n. 2, p. 274–281, 2003. ISSN 10459227. Citado na página 20.

HUANG, G.-B.; ZHOU, H.; DING, X.; ZHANG, R. Extreme Learning Machine for Regression and Multiclass Classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, v. 42, n. 2, p. 513–529, 2012. ISSN 1083-4419. Disponível em: <<http://www.ncbi.nlm.nih.gov/pubmed/21984515><http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6035797>>. Citado 3 vezes nas páginas 20, 35 e 51.

HUANG, G.-B.; ZHU, Q.-y.; SIEW, C.-k. Extreme learning machine: Theory and applications. *Neurocomputing*, v. 70, n. 1-3, p. 489–501, 2006. ISSN 09252312. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0925231206000385>>. Citado 2 vezes nas páginas 20 e 33.

IMAN, R. L.; DAVENPORT, J. M. Approximations of the critical region of the fbietkan statistic. *Communications in Statistics - Theory and Methods*, v. 9, n. 6, p. 571–595, jan 1980. ISSN 0361-0926. Disponível em: <<http://www.tandfonline.com/doi/abs/10.1080/03610928008827904>>. Citado na página 111.

INABA, F. K.; SALLES, E. O. T.; PERRON, S.; CAPOROSI, G. DGR-ELM-Distributed Generalized Regularized ELM for classification. *Neurocomputing*, Elsevier B.V., v. 275, p. 1522–1530, 2017. ISSN 18728286. Disponível em: <<https://doi.org/10.1016/j.neucom.2017.09.090>>. Citado na página 38.

JIANG, M.; PAN, Z.; LI, N. Multi-label text categorization using L21-norm minimization extreme learning machine. *Neurocomputing*, v. 261, p. 4–10, 2017. ISSN 18728286. Citado 2 vezes nas páginas 21 e 22.

KOCEV, D.; DŽEROSKI, S.; WHITE, M. D.; NEWELL, G. R.; GRIFFIOEN, P. Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition. *Ecological Modelling*, v. 220, n. 8, p. 1159–1168, apr 2009. ISSN 03043800. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0304380009000775>>. Citado na página 43.

KONGSOROT, Y.; HORATA, P.; SUNAT, K. *Applying Regularization Least Squares Canonical Correlation Analysis in Extreme Learning Machine for Multi-label Classification Problems*. Cham: Springer International Publishing, 2015. v. 4. 377–396 p. (Proceedings in Adaptation, Learning and Optimization, v. 4). ISSN 978-3-319-14065-0. ISBN 978-3-319-14065-0. Disponível em: <<http://link.springer.com/10.1007/978-3-319-14066-7>>. Citado 2 vezes nas páginas 21 e 22.

LI, K.; PENG, J.-X.; IRWIN, G. A fast nonlinear model identification method. *IEEE Transactions on Automatic Control*, v. 50, n. 8, p. 1211–1216, aug 2005. ISSN 0018-9286. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1492567>><<http://ieeexplore.ieee.org/document/1492567/>>. Citado 2 vezes nas páginas 19 e 20.

MARTÍNEZ-MARTÍNEZ, J. M.; ESCANDELL-MONTERO, P.; SORIA-OLIVAS, E.; MARTÍN-GUERRERO, J. D.; MAGDALENA-BENEDITO, R.; GÓMEZ-SANCHIS, J. Regularized extreme learning machine for regression problems. *Neurocomputing*, v. 74, n. 17, p. 3716–3721, 2011. ISSN 09252312. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S092523121100378X>>. Citado 3 vezes nas páginas 20, 35 e 36.

MCCULLOCH, W. S.; PITTS, W. A Logical Calculus of the Idea Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, v. 5, p. 115–133, 1943. ISSN 0007-4985. Disponível em: <<http://www.cse.chalmers.se/~coquand/AUTOMATA/mcp.p>>. Citado na página 19.

MEYER, C. D. *Matrix Analysis and Applied Linear Algebra*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2000. ISBN 0-89871-454-0. Citado na página 109.

MINSKY, M.; PAPERT, S. *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA, USA: MIT Press, 1969. Citado na página 19.

NEMENYI, P. *Distribution-free multiple comparisons*. Tese (Ph.D.) — Princeton University, 1963. Citado na página 111.

RAO, C. R.; MITRA, S. K. *Generalized Inverse of Matrices and Its Applications*. [S.l.]: Wiley, 1971. (Probability and Statistics Series). ISBN 9780471708216. Citado na página 34.

ROCKAFELLAR, R. *Convex Analysis*. [S.l.]: Princeton landmarks in mathematics and physics, 1970. ISBN 9780691015866. Citado na página 99.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386–408, 1958. ISSN 1939-1471. Citado na página 19.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, v. 323, n. 6088, p. 533–536, oct 1986. ISSN 0028-0836. Disponível em: <<http://www.nature.com/doifinder/10.1038/323533a0>>. Citado na página 19.

SHESKIN, D. J. *Handbook of Parametric and Nonparametric Statistical Procedures*. 4. ed. [S.l.]: Chapman & Hall/CRC, 2000. ISBN 9781584888147. Citado na página 111.

SIMILÄ, T.; TIKKA, J. Input selection and shrinkage in multiresponse linear regression. *Computational Statistics & Data Analysis*, v. 52, n. 1, p. 406–422, sep 2007. ISSN 01679473. Disponível em: <<http://linkinghub.elsevier.com/retrieve/pii/S0167947307000205>>. Citado na página 22.

SPYROMITROS-XIOUFIS, E.; TSOUMAKAS, G.; GROVES, W.; VLAHAVAS, I. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, Springer US, v. 104, n. 1, p. 55–98, 2016. ISSN 15730565. Citado 4 vezes nas páginas 22, 25, 43 e 59.

STARCK, J.-L.; MURTAGH, F.; FADILI, J. M. *Sparse Image and Signal Processing: Wavelets, Curvelets, Morphological Diversity*. New York, NY, USA: Cambridge University Press, 2010. ISBN 978-0-521-11913-9. Citado na página 27.

SUN, X.; WANG, J.; JIANG, C.; XU, J.; FENG, J. ELM-ML: Study on Multi-label Classification Using Extreme Learning Machine. v. 6, 2016. Disponível em: <<http://link.springer.com/10.1007/978-3-319-28397-5>>. Citado 2 vezes nas páginas 21 e 22.

TAMURA, S.; TATEISHI, M. Capabilities of a four-layered feedforward neural network: Four layers versus three. *IEEE Transactions on Neural Networks*, v. 8, n. 2, p. 251–255, 1997. ISSN 10459227. Citado na página 20.

TIBSHIRANI, R. Regression Selection and Shrinkage via the Lasso. *Journal of the Royal Statistical Society B (Methodological)*, v. 58, n. 1, p. 267–288, 1996. ISSN 00359246. Disponível em: <<http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.7574>>. Citado na página 35.

WANG, N.; ER, M. J.; HAN, M. Parsimonious extreme learning machine using recursive orthogonal least squares. *IEEE Transactions on Neural Networks and Learning Systems*, v. 25, n. 10, p. 1828–1841, 2014. ISSN 21622388. Disponível em:

<<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6704311>>. Citado na página 20.

WANG, N.; ER, M. J.; HAN, M. Generalized Single-Hidden Layer Feedforward Networks for Regression Problems. *IEEE Transactions on Neural Networks and Learning Systems*, v. 26, n. 6, p. 1161–1176, jun 2015. ISSN 2162-237X. Disponível em: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6856201>>. Citado na página 20.

WANG, Y.; DOU, Y.; LIU, X.; LEI, Y. PR-ELM: Parallel regularized extreme learning machine based on cluster. *Neurocomputing*, Elsevier, v. 173, p. 1073–1081, jan 2016. ISSN 09252312. Disponível em: <<http://dx.doi.org/10.1016/j.neucom.2015.08.066http://linkinghub.elsevier.com/retrieve/pii/S0925231215012461>>. Citado 3 vezes nas páginas 20, 21 e 51.

WANG, Y.; WIPF, D.; LING, Q.; CHEN, W.; WASSELL, I. Multi-Task Learning for Subspace Segmentation. *Proceedings of the 32nd International Conference on Machine Learning*, v. 37, p. 1209–1217, 2015. Disponível em: <<http://proceedings.mlr.press/v37/wangc15.html>>. Citado na página 43.

WERBOS, P. J. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Tese (Doutorado) — Harvard University, 1974. Citado na página 19.

WILAMOWSKI, B. M.; YU, H. Neural network learning without backpropagation. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, v. 21, n. 11, p. 1793–1803, 2010. ISSN 1941-0093. Disponível em: <<http://www.ncbi.nlm.nih.gov/pubmed/20858577>>. Citado 2 vezes nas páginas 19 e 20.

WILCOXON, F. Individual Comparisons of Grouped Data by Ranking Methods. *Journal of Economic Entomology*, v. 39, n. 2, p. 269–270, apr 1945. ISSN 1938-291X. Disponível em: <<http://academic.oup.com/jee/article/39/2/269/2203647/Individual-Comparisons-of-Grouped-Data-by-Ranking>>. Citado na página 112.

WRIGHT, J.; YANG, A. Y.; GANESH, A.; SASTRY, S. S.; MA, Y. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 31, n. 2, p. 210–227, 2009. ISSN 01628828. Citado na página 28.

XU, W.; WANG, M.; CAI, J.-F.; TANG, A. Sparse Error Correction From Nonlinear Measurements With Applications in Bad Data Detection for Power Networks. *IEEE Transactions on Signal Processing*, v. 61, n. 24, p. 6175–6187, 2013. ISSN 1053-587X. Disponível em: <<http://ieeexplore.ieee.org/document/6603360/>>. Citado 2 vezes nas páginas 28 e 44.

YAO, X. A review of evolutionary artificial neural networks. *International Journal of Intelligent Systems*, v. 8, n. 4, p. 539–567, 1993. ISSN 08848173. Disponível em: <<http://doi.wiley.com/10.1002/int.4550080406>>. Citado 2 vezes nas páginas 19 e 20.

ZAR, J. H. *Biostatistical Analysis*. 4. ed. [S.l.]: Prentice Hall, 1998. ISBN 9780130815422. Citado na página 111.

ZHANG, K.; LUO, M. Outlier-robust extreme learning machine for regression problems. *Neurocomputing*, Elsevier, v. 151, p. 1519–1527, mar 2015. ISSN 09252312. Disponível em:

<<http://linkinghub.elsevier.com/retrieve/pii/S0925231214012053>>. Citado 6 vezes nas páginas 21, 28, 36, 43, 44 e 61.

ZHANG, L.; ZHANG, D. *A High Speed Multi-label Classifier Based on Extreme Learning Machines*. Cham: Springer International Publishing, 2016. v. 6. 437–455 p. (Proceedings in Adaptation, Learning and Optimization, v. 6). ISBN 978-3-319-28396-8. Disponível em: <<http://link.springer.com/10.1007/978-3-319-28397-5><http://arxiv.org/abs/1506.02509>>. Citado 2 vezes nas páginas 21 e 22.

ZHANG, N.; DING, S.; ZHANG, J. Multi layer ELM-RBF for multi-label learning. *Applied Soft Computing Journal*, Elsevier B.V., v. 43, p. 535–545, 2016. ISSN 15684946. Disponível em: <<http://dx.doi.org/10.1016/j.asoc.2016.02.039>>. Citado 2 vezes nas páginas 21 e 22.

ZHANG, Y.; YEUNG, D.-y. A Convex Formulation for Learning Task Relationships in Multi-Task Learning. *Uai*, p. 733–442, 2010. ISSN 0031-8655. Citado na página 43.

ZHEN, X.; WANG, Z.; ISLAM, A.; BHADURI, M.; CHAN, I.; LI, S. Multi-scale deep networks and regression forests for direct bi-ventricular volume estimation. *Medical Image Analysis*, Elsevier B.V., v. 30, p. 120–129, 2016. ISSN 13618423. Disponível em: <<http://dx.doi.org/10.1016/j.media.2015.07.003>>. Citado na página 43.

ZHEN, X.; YU, M.; HE, X.; LI, S. Multi-Target Regression via Robust Low-Rank Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 8828, n. c, p. 1–1, 2017. ISSN 0162-8828. Disponível em: <<http://ieeexplore.ieee.org/document/7888599/>>. Citado 2 vezes nas páginas 22 e 43.

ZOU, H.; HASTIE, T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, v. 67, n. 2, p. 301–320, apr 2005. ISSN 1369-7412. Disponível em: <<http://doi.wiley.com/10.1111/j.1467-9868.2005.00503.x>>. Citado na página 35.

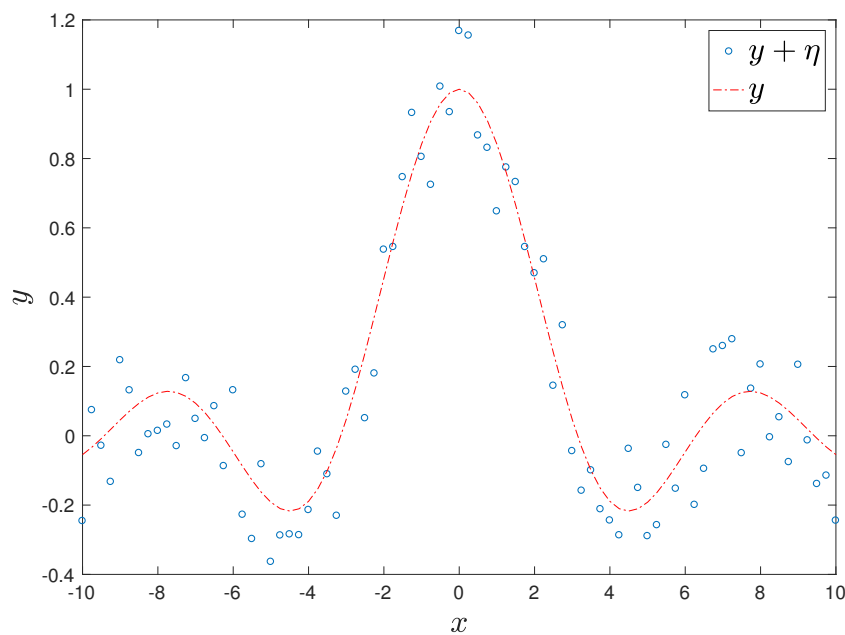
Apêndices

APÊNDICE A – Conexões entre regularização, *overfitting* e esparsidade em ELM

No Capítulo 2, uma revisão de esparsidade, regularizações ℓ_1 e ℓ_2 e ELM é feita. Em especial, na seção 2.2 realiza-se uma conexão entre a norma ℓ_1 e robustez à *outliers* para facilitar o entendimento do OR-ELM. Nesse caso, a norma ℓ_1 é aplicada ao termo do erro, conforme descrito pela Equação 2.7. Entretanto, no R-ELM, a regularização ocorre nos pesos de saída, na tentativa de se obter uma melhor generalização, isto é, reduzir o *overfitting*. Assim, uma dúvida natural surge: Por que a regularização em ELM melhora generalização da rede resultante?

Para elucidar essa pergunta, considere o problema de regressão onde deseja-se ajustar os dados exibidos na Figura A.1.

Figura A.1 – Dados usados para exemplificar o efeito da regularização.



Fonte: Produção do próprio autor.

Os dados foram obtidos de uma função *sinc* conforme

$$y_i = \begin{cases} \sin(x_i)/x_i, & \text{se } x_i \neq 0 \\ 1, & \text{se } x_i = 0 \end{cases}, \quad (\text{A.1})$$

para x_i no intervalo $[-10, 10]$, e $i = 1, 2, \dots, 81$. Os dados foram contaminados por um ruído uniforme, $\boldsymbol{\eta}$, entre $[-0.2, 0.2]$,

$$\mathbf{y}' = \mathbf{y} + \boldsymbol{\eta}. \quad (\text{A.2})$$

Agora, será utilizado uma rede SLFN para estimar a saída \mathbf{y} , com treinamento feito pelo ELM. Assim, deseja-se resolver o seguinte problema

$$\mathbf{y}' = \mathbf{H}\boldsymbol{\beta} + \boldsymbol{\eta}, \quad (\text{A.3})$$

para \mathbf{H} definida na Equação 2.34, de modo que a saída da rede neural, $\hat{\mathbf{y}} = \mathbf{H}\boldsymbol{\beta}$, se aproxime da saída sem ruído, \mathbf{y} .

Primeiro, avalia-se o treinamento da rede pelo ELM sem regularização, usando 1, 2, 5, 10, 15 e 20 neurônios (\widetilde{N}). A Figura A.2 resume os gráficos obtidos na aproximação usando a rede neural. Note que à medida que o número de neurônios aumenta, a complexidade do que se é aprendido pela rede aumenta também, e com isso o efeito do *overfitting* aumenta. Além disso, os valores dos elementos de $\boldsymbol{\beta}$ aumentam em magnitude. Isso pode ser observado pela Tabela A.1 que resume as normas ℓ_2 e ℓ_1 obtidas nesse teste. Note que

Tabela A.1 – Norma ℓ_1 e ℓ_2 dos pesos de saída, $\boldsymbol{\beta}$, para 1, 2, 5, 10, 15, e 20 neurônios.

\widetilde{N}	1	2	5	10	15	20
$\ \boldsymbol{\beta}\ _2^2$	0.0473	57.971	106.83	7.6529×10^5	1.857×10^8	3.2279×10^{13}
$\ \boldsymbol{\beta}\ _1$	0.2175	10.766	17.92	1913.5	25023	1.4568×10^7

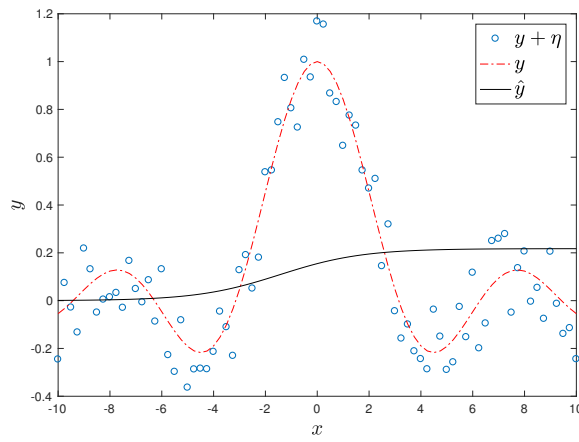
os pesos de saída, à medida que a rede fica mais complexa, aumentam consideravelmente. Portanto, ao limitar a magnitude dos pesos de saída, através da regularização, tem-se que, em geral, a rede resultante possui maior capacidade de generalização.

Com 20 neurônios, percebe-se que o *overfitting* já está presente na rede resultante. Mantendo-se os mesmo 20 neurônios, agora realiza-se a regularização dos pesos de saída, usando a norma ℓ_2 , isto é, o R-ELM com regularização *ridge*. Portanto, tem-se o seguinte problema

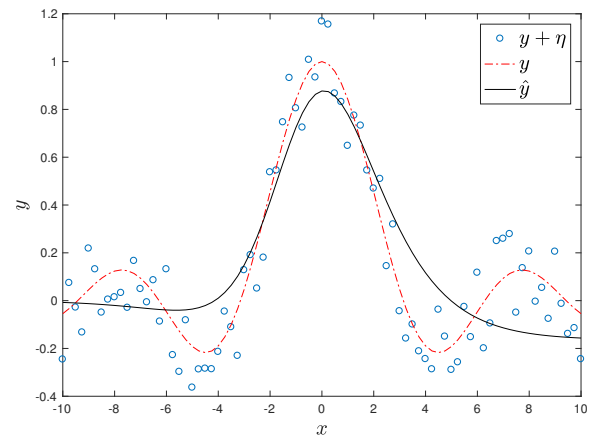
$$\begin{aligned} &\underset{\boldsymbol{\beta}}{\text{minimizar}} \quad \frac{1}{2} \|\boldsymbol{\beta}\|_2^2 + \frac{C}{2} \|\boldsymbol{\eta}\|_2^2 \\ &\text{sujeito a} \quad \mathbf{H}\boldsymbol{\beta} - \mathbf{y} = \boldsymbol{\eta}. \end{aligned} \quad (\text{A.4})$$

Neste caso, os valores de 0.01, 1, 10, 100, 10000, e 100000 foram utilizado para o parâmetro C . A Figura A.3 exibe a aproximação obtida para os diversos valores de C testados.

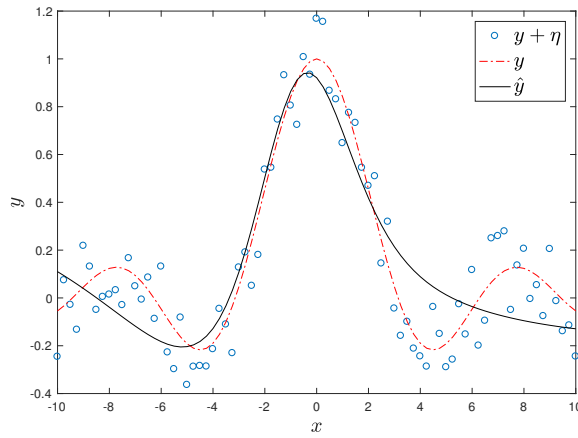
Figura A.2 – Aproximação obtida por rede neural treinada pelo ELM sem regularização e com (a) 1, (b) 2, (c) 5, (d) 10, (e) 15 e (f) 20 neurônios.



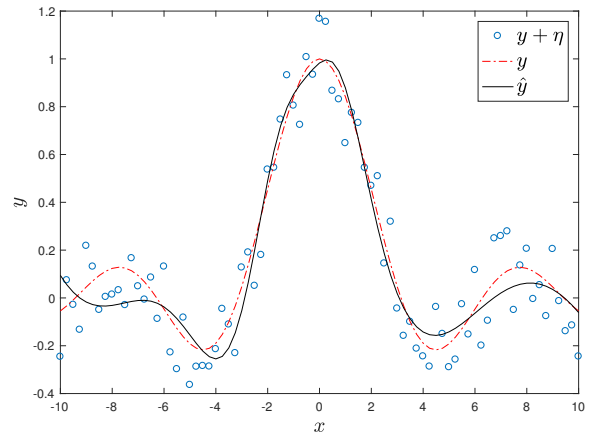
(a) 1 neurônio



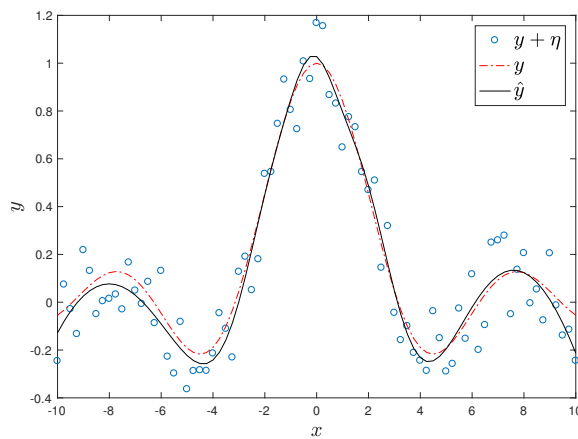
(b) 2 neurônios



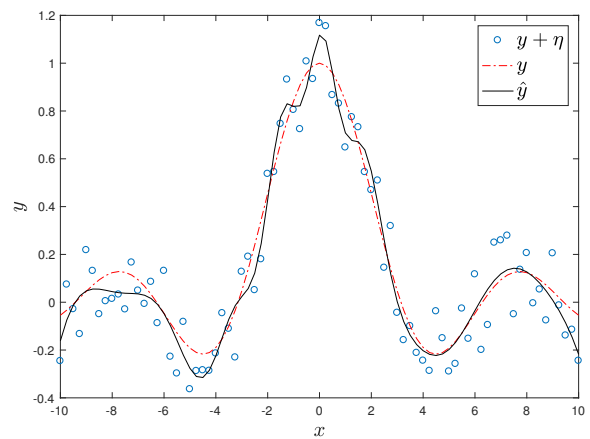
(c) 5 neurônios



(d) 10 neurônios



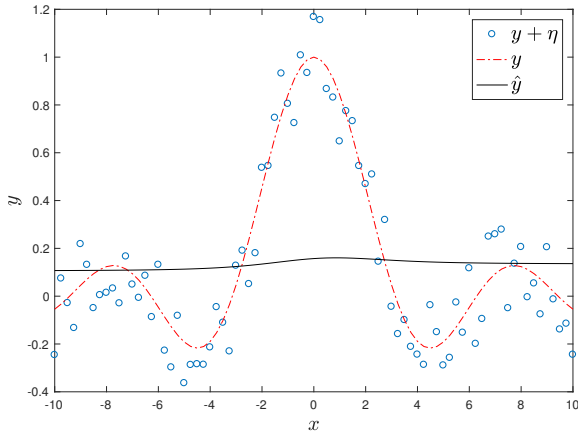
(e) 15 neurônios



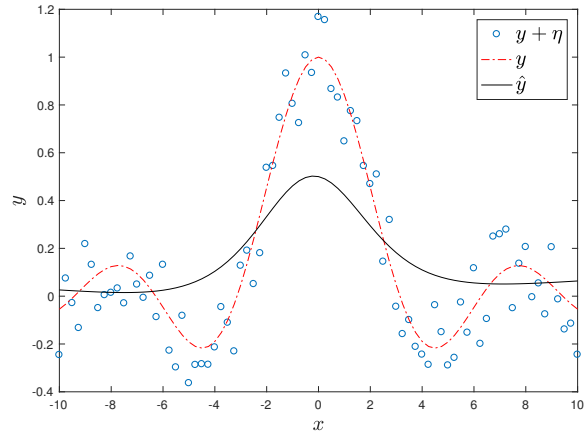
(f) 20 neurônios

Fonte: Produção do próprio autor.

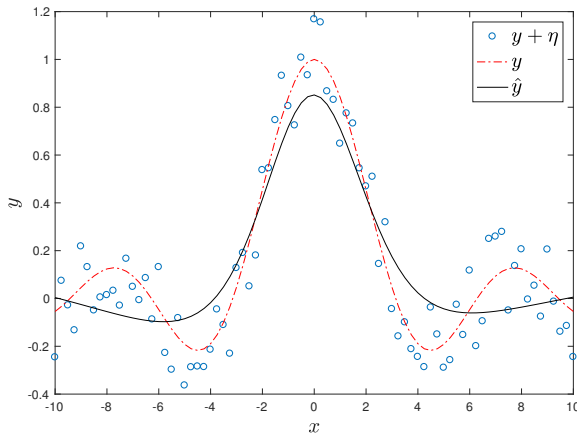
Figura A.3 – Aproximação obtida por rede neural treinada pelo ELM com regularização *ridge* e parâmetro C igual a (a) 0.01, (b) 1, (c) 10, (d) 100, (e) 10000 e (f) 100000.



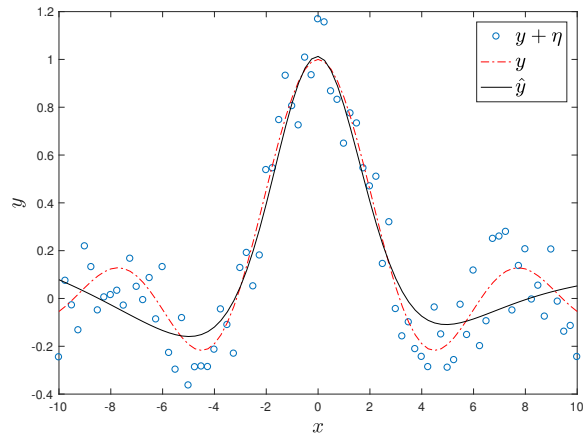
(a) $C = 0.01$



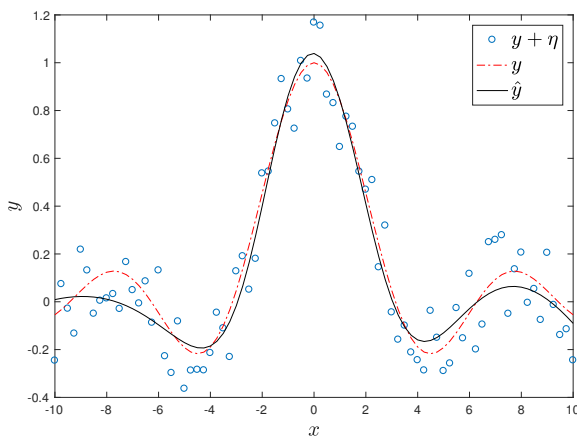
(b) $C = 1$



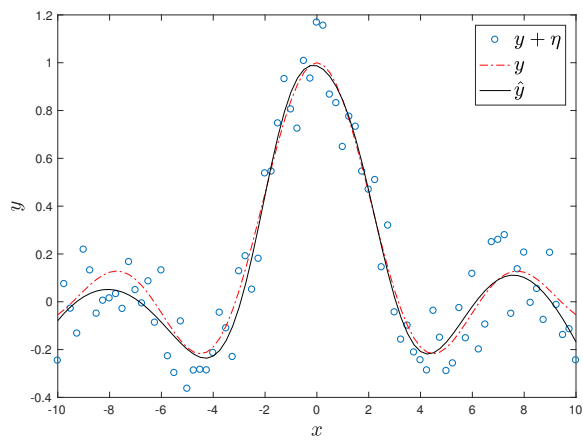
(c) $C = 10$



(d) $C = 100$



(e) $C = 10000$



(f) $C = 100000$

Fonte: Produção do próprio autor.

Note que, de maneira similar ao caso sem regularização, foi possível controlar a complexidade aprendida pela rede apenas variando o parâmetro C . Assim, o efeito do *overfitting* foi atenuado, mesmo com 20 neurônios, apenas limitando a magnitude de β . A Tabela A.2 resume os valores da norma ℓ_2 e ℓ_1 para os diferentes valores de C .

Tabela A.2 – Norma ℓ_1 e ℓ_2 dos pesos de saída, β , para C igual a 0.01, 1, 10, 100, 10000, e 100000.

C	0.01	1	10	100	10000	100000
$\ \beta\ _2^2$	0.0037715	2.053	11.33	31.837	1432	6651.7
$\ \beta\ _1$	0.23945	54.769	13.023	19.949	137.61	297.51

À medida que C aumenta, a norma ℓ_2 e ℓ_1 aumentam também. Contudo, isso ocorre de modo menos agressivo quando comparado ao caso sem regularização. Note que quanto maior o valor de C , menos importância tem a norma ℓ_2 dos pesos de saída no problema de otimização. Por isso, quando aumenta-se o valor de C maior é a magnitude dos pesos de saída.

Mantendo-se os 20 neurônios, na camada oculta, realiza-se a regularização dos pesos de saída usando a norma ℓ_1 . Considere o seguinte problema

$$\begin{aligned} & \underset{\beta}{\text{minimizar}} && \frac{\lambda}{2} \|\beta\|_1 + \frac{1}{2} \|\eta\|_2^2 \\ & \text{sujeito a} && \mathbf{H}\beta - \mathbf{y} = \eta. \end{aligned} \tag{A.5}$$

Os seguintes valores de λ foram testados: 1, 0.1, 0.01, 0.001, 0.0001, e 0.00001. A Figura A.4 exibe a aproximação obtida, com regularização ℓ_1 , para os diversos valores de λ testados.

Novamente, nota-se que, mesmo com 20 neurônios, a rede neural resultante do treinamento pelo ELM com regularização ℓ_1 apresentou uma melhor generalização quando comparada com a rede treinada pelo ELM sem regularização. À medida que o valor de λ diminui, observa-se que a rede consegue aprender padrões mais complexos. Isso é ratificado ao mensurar a quantidade de β_i iguais zero.

A Tabela A.3 resume os valores da norma ℓ_1 e ℓ_2 dos pesos de saída, além do número de β_i iguais a zero, para λ igual 1, 0.1, 0.01, 0.001, 0.0001 e 0.00001.

Tabela A.3 – Norma ℓ_1 e ℓ_2 dos pesos de saída, β , para λ igual a 1, 0.1, 0.01, 0.001, 0.00001, e 0.000001.

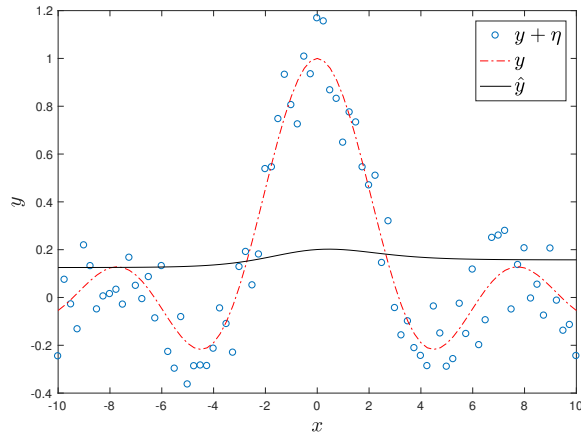
C	1	0.1	0.01	0.001	0.00001	0.000001
$\ \beta\ _2^2$	0.04044	25.969	80.816	2204.2	7577	8430.5
$\ \beta\ _1$	0.28259	10.944	20.088	134.53	296.48	332.41
$\# \{\beta_i = 0\}$	18	14	13	10	2	0

Observa-se que à medida que λ diminui, reduz-se a contribuição da regularização pela norma ℓ_1 dos pesos de saída. Com isso, o número de elementos iguais a zero em β diminui (fica menos esperso), e a norma ℓ_1 e ℓ_2 aumenta.

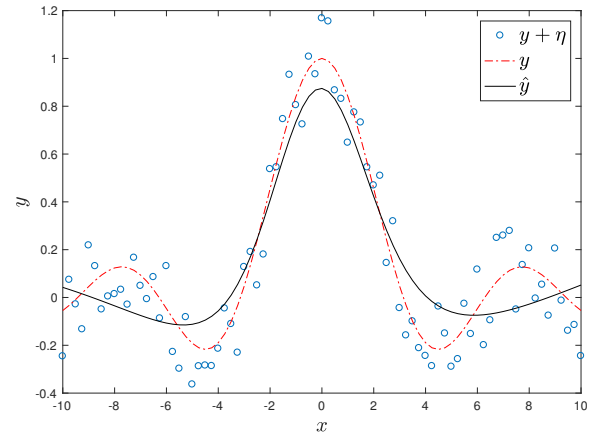
Assim, as seguintes observações podem ser feitas

- o número de neurônios da rede neural, quando treinados pelo ELM sem regularização, deve ser bem escolhido. Um número excessivo de neurônios irá acarretar na situação de *overfitting*, diminuindo o poder de generalização da rede;
- a regularização pela norma ℓ_2 é capaz de diminuir o efeito do *overfitting*. Nessa regularização, a magnitude dos pesos são limitados, o que ajuda a evitar o *overfitting*; e
- a regularização pela norma ℓ_1 resulta em um vetor de pesos, β , esperso. Isso diminui a complexidade da rede, tornando-a mais compacta, e com isso, ajuda a evitar o *overfitting*.

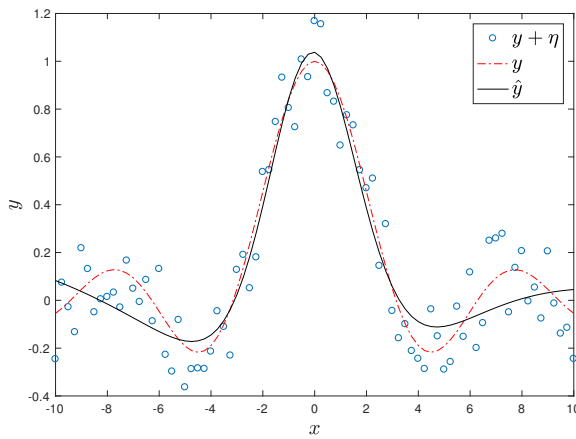
Figura A.4 – Aproximação obtida por rede neural treinada pelo ELM com regularização ℓ_1 e parâmetro λ igual a (a) 1, (b) 0.1, (c) 0.01, (d) 0.001, (e) 0.0001 e (f) 0.00001.



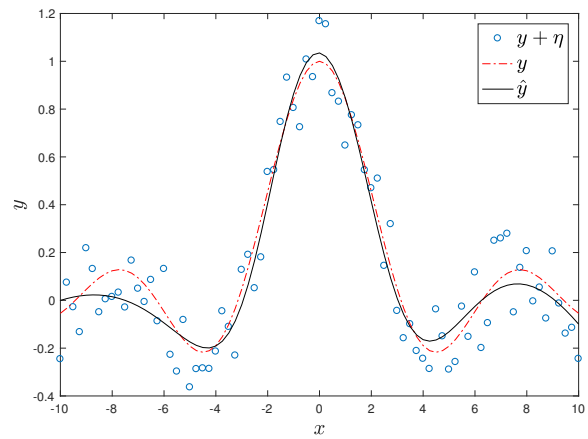
(a) $\lambda = 1$



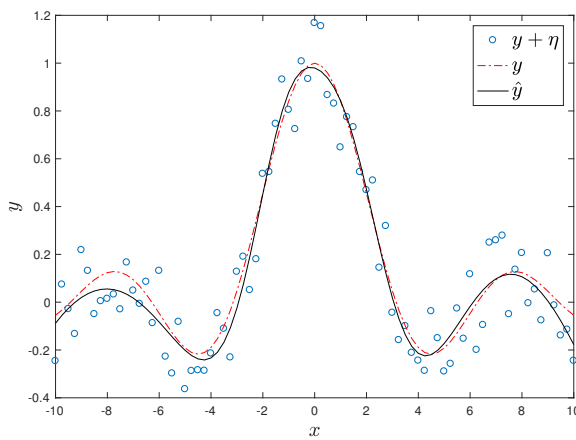
(b) $\lambda = 0.1$



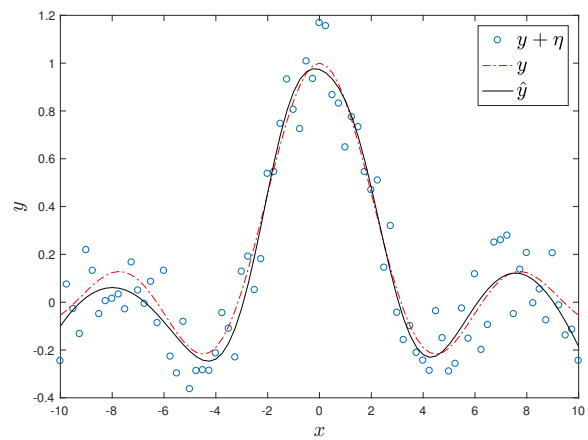
(c) $\lambda = 0.01$



(d) $\lambda = 0.001$



(e) $\lambda = 0.0001$



(f) $\lambda = 0.00001$

Fonte: Produção do próprio autor.

APÊNDICE B – *Soft-Thresholding Operator*

Considere o seguinte problema

$$x^+ = \arg \min |x| + \frac{\lambda}{2}(x - \nu)^2 \quad (\text{B.1})$$

com x e $\nu \in \mathbb{R}$ e $\lambda > 0$. Embora o primeiro termo não seja diferenciável, é possível obter uma solução fechada para esse problema usando cálculo subdiferencial (para maior aprofundamento no assunto ver (ROCKAFELLAR, 1970; BOYD, 2010)). A solução de Equação B.1 é dada por

$$x^+ = S_\lambda(x), \quad (\text{B.2})$$

onde S é o operador *soft-thresholding* e é definido como

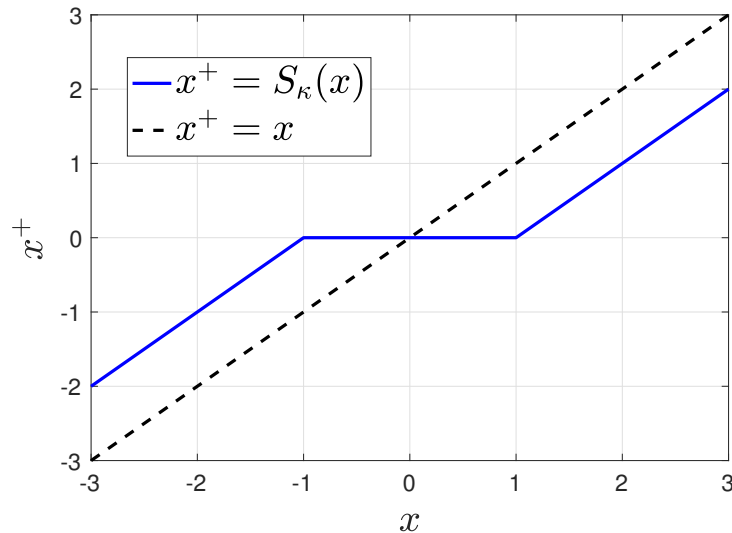
$$S_\kappa(a) = \begin{cases} a - \kappa, & \text{se } a > \kappa \\ 0, & \text{se } |a| \leq \kappa \\ a + \kappa, & \text{se } a < -\kappa \end{cases} \quad (\text{B.3})$$

ou de forma equivalente,

$$S_\kappa(a) = \left(1 - \frac{\kappa}{|a|}\right)_+ a \quad (\text{B.4})$$

para $a \neq 0$, onde $(a)_+ = \max(0, a)$. A Figura B.1 exibe o resultado da função Equação B.4 no intervalo de $[-3, 3]$ e $\kappa = 1$.

Figura B.1 – *Soft-thresholding operator* para $\kappa = 1$.



Fonte: Produção do próprio autor.

Note que para um valor de x no intervalo $[-\kappa, \kappa]$ o operador *soft-thresholding* encolhe (*shrink*) x para zero. Para valores de $|x| > \kappa$ o operador move x , por uma quantidade κ , em direção à origem.

APÊNDICE C – Derivações Matemáticas das Atualizações do ADMM

Forma Escalada

Nesta seção apresenta-se o desenvolvimento matemático para obtenção da Equação 3.4 a partir da Equação 3.5.

Proposição 1. *Seja $\mathbf{R} = \mathbf{B} - \mathbf{Z}$, então,*

$$\frac{\rho}{2} \|\mathbf{R}\|_F^2 + \langle \mathbf{Y}, \mathbf{R} \rangle = \frac{\rho}{2} \|\mathbf{R} + \mathbf{U}\|_F^2 - \frac{\rho}{2} \|\mathbf{U}\|_F^2 \quad (\text{C.1})$$

onde $\mathbf{U} = (1/\rho)\mathbf{Y}$.

Demonstração. A partir do lado esquerdo da Equação C.1 tem-se

$$\begin{aligned} \frac{\rho}{2} \|\mathbf{R}\|_F^2 + \langle \mathbf{Y}, \mathbf{R} \rangle &= \frac{\rho}{2} \text{tr}(\mathbf{R}^\top \mathbf{R}) + \langle \mathbf{Y}, \mathbf{R} \rangle \\ &= \frac{\rho}{2} \text{tr}(\mathbf{R}^\top \mathbf{R}) + \langle \mathbf{Y}, \mathbf{R} \rangle + \frac{1}{2\rho} \text{tr}(\mathbf{Y}^\top \mathbf{Y}) - \frac{1}{2\rho} \text{tr}(\mathbf{Y}^\top \mathbf{Y}) \\ &= \frac{\rho}{2} \text{tr}(\mathbf{R}^\top \mathbf{R}) + \text{tr}(\mathbf{Y}^\top \mathbf{R}) + \frac{1}{2\rho} \text{tr}(\mathbf{Y}^\top \mathbf{Y}) - \frac{1}{2\rho} \text{tr}(\mathbf{Y}^\top \mathbf{Y}) \\ &= \frac{\rho}{2} \text{tr} \left(\mathbf{R}^\top \mathbf{R} + \frac{2}{\rho} \mathbf{Y}^\top \mathbf{R} + \frac{1}{\rho^2} \mathbf{Y}^\top \mathbf{Y} \right) - \frac{1}{2\rho} \text{tr}(\mathbf{Y}^\top \mathbf{Y}) \\ &= \frac{\rho}{2} \text{tr} \left(\left(\mathbf{R} + \frac{1}{\rho} \mathbf{Y} \right)^\top \left(\mathbf{R} + \frac{1}{\rho} \mathbf{Y} \right) \right) - \frac{1}{2\rho} \text{tr}(\mathbf{Y}^\top \mathbf{Y}) \\ &= \frac{\rho}{2} \left\| \mathbf{R} + \frac{1}{\rho} \mathbf{Y} \right\|_F^2 - \frac{1}{2\rho} \text{tr}(\mathbf{Y}^\top \mathbf{Y}) \\ &= \frac{\rho}{2} \left\| \mathbf{R} + \frac{1}{\rho} \mathbf{Y} \right\|_F^2 - \frac{1}{2\rho} \text{tr} \left(\frac{1}{\rho} \mathbf{Y}^\top \mathbf{Y} \frac{1}{\rho} \right) \rho^2 \\ &= \frac{\rho}{2} \|\mathbf{R} + \mathbf{U}\|_F^2 - \frac{\rho}{2} \text{tr}(\mathbf{U}^\top \mathbf{U}) \\ &= \frac{\rho}{2} \|\mathbf{R} + \mathbf{U}\|_F^2 - \frac{\rho}{2} \|\mathbf{U}\|_F^2 \end{aligned}$$

□

Note que para a obtenção da Equação 3.26 a partir da Equação 3.25 segue exatamente a demonstração de Equação C.1, bastando fazer $\mathbf{R} = \widetilde{\mathbf{A}}\mathbf{E} + \widetilde{\mathbf{B}}\mathbf{B} + \widetilde{\mathbf{C}}\mathbf{Z} + \widetilde{\mathbf{D}}$.

Atualização de \mathbf{B}

Proposição 2. *Seja $g = \|\mathbf{HB} - \mathbf{T}'\|_F^2$, então*

$$\frac{\partial g}{\partial \mathbf{B}} = 2\mathbf{H}^\top (\mathbf{HB} - \mathbf{T}') \quad (\text{C.2})$$

Demonstração.

$$\begin{aligned} \frac{\partial}{\partial \mathbf{B}} \|\mathbf{HB} - \mathbf{T}'\|_F^2 &= \frac{\partial}{\partial \mathbf{B}} \text{tr} \left((\mathbf{HB} - \mathbf{T}')^\top (\mathbf{HB} - \mathbf{T}') \right) \\ &= \frac{\partial}{\partial \mathbf{B}} \text{tr} \left((\mathbf{HB})^\top \mathbf{HB} - (\mathbf{HB})^\top \mathbf{T}' - \mathbf{T}'^\top \mathbf{HB} + \mathbf{T}'^\top \mathbf{T}' \right) \\ &= \frac{\partial}{\partial \mathbf{B}} \left(\text{tr} \left((\mathbf{HB})^\top \mathbf{HB} \right) - \text{tr} \left((\mathbf{HB})^\top \mathbf{T}' \right) - \text{tr} \left(\mathbf{T}'^\top \mathbf{HB} \right) + \text{tr} \left(\mathbf{T}'^\top \mathbf{T}' \right) \right) \\ &= \frac{\partial}{\partial \mathbf{B}} \text{tr} \left((\mathbf{HB})^\top \mathbf{HB} \right) - \frac{\partial}{\partial \mathbf{B}} \text{tr} \left((\mathbf{HB})^\top \mathbf{T}' \right) - \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{T}'^\top \mathbf{HB} \right) + \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{T}'^\top \mathbf{T}' \right) \\ &= \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{B}^\top \mathbf{H}^\top \mathbf{HB} \right) - \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{B}^\top \mathbf{H}^\top \mathbf{T}' \right) - \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{T}'^\top \mathbf{HB} \right) + \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{T}'^\top \mathbf{T}' \right) \\ &= \mathbf{H}^\top \mathbf{HB} + \mathbf{H}^\top \mathbf{HB} - \mathbf{H}^\top \mathbf{T}' - \mathbf{H}^\top \mathbf{T}' + \mathbf{0} \\ &= 2\mathbf{H}^\top \mathbf{HB} - 2\mathbf{H}^\top \mathbf{T}' \\ &= 2\mathbf{H}^\top (\mathbf{HB} - \mathbf{T}') \end{aligned}$$

□

Para o GR-ELM $\mathbf{T}' = \mathbf{T}$ e para GOR-ELM $\mathbf{T}' = \mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k$.

Proposição 3. *Seja $g = \|\mathbf{B} + (\mathbf{U} - \mathbf{Z})\|_F^2$, então*

$$\frac{\partial g}{\partial \mathbf{B}} = 2(\mathbf{B} + (\mathbf{U} - \mathbf{Z})) \quad (\text{C.3})$$

Demonstração.

$$\begin{aligned} \frac{\partial g}{\partial \mathbf{B}} &= \frac{\partial}{\partial \mathbf{B}} \|\mathbf{B} + (\mathbf{U} - \mathbf{Z})\|_F^2 \\ &= \frac{\partial}{\partial \mathbf{B}} \text{tr} \left((\mathbf{B} + (\mathbf{U} - \mathbf{Z}))^\top (\mathbf{B} + (\mathbf{U} - \mathbf{Z})) \right) \\ &= \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{B}^\top \mathbf{B} + \mathbf{B}^\top (\mathbf{U} - \mathbf{Z}) + (\mathbf{U} - \mathbf{Z})^\top \mathbf{B} + (\mathbf{U} - \mathbf{Z})^\top (\mathbf{U} - \mathbf{Z}) \right) \\ &= \frac{\partial}{\partial \mathbf{B}} \left(\text{tr} \left(\mathbf{B}^\top \mathbf{B} \right) + \text{tr} \left(\mathbf{B}^\top (\mathbf{U} - \mathbf{Z}) \right) + \text{tr} \left((\mathbf{U} - \mathbf{Z})^\top \mathbf{B} \right) + \text{tr} \left((\mathbf{U} - \mathbf{Z})^\top (\mathbf{U} - \mathbf{Z}) \right) \right) \\ &= \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{B}^\top \mathbf{B} \right) + \frac{\partial}{\partial \mathbf{B}} \text{tr} \left(\mathbf{B}^\top (\mathbf{U} - \mathbf{Z}) \right) + \frac{\partial}{\partial \mathbf{B}} \text{tr} \left((\mathbf{U} - \mathbf{Z})^\top \mathbf{B} \right) + \frac{\partial}{\partial \mathbf{B}} \text{tr} \left((\mathbf{U} - \mathbf{Z})^\top (\mathbf{U} - \mathbf{Z}) \right) \\ &= 2\mathbf{B} + (\mathbf{U} - \mathbf{Z}) + (\mathbf{U} - \mathbf{Z}) + \mathbf{0} \\ &= 2(\mathbf{B} + (\mathbf{U} - \mathbf{Z})) \end{aligned}$$

□

Fazendo o gradiente da função objetivo, Equação 3.6, com respeito a \mathbf{B} igual a $\mathbf{0}$ (i.e., $\nabla_{\mathbf{B}} L(\mathbf{B}, \mathbf{Z}^k, \mathbf{U}^k) = \mathbf{0}$)

$$\begin{aligned}
C\mathbf{H}^\top(\mathbf{H}\mathbf{B} - \mathbf{T}) + \lambda_2\mathbf{B} + \rho(\mathbf{B} - \mathbf{Z}^k + \mathbf{U}^k) &= \mathbf{0} \\
\mathbf{H}^\top(\mathbf{H}\mathbf{B} - \mathbf{T}) + \frac{\lambda_2}{C}\mathbf{B} + \frac{\rho}{C}(\mathbf{B} - \mathbf{Z}^k + \mathbf{U}^k) &= \mathbf{0} \\
\mathbf{H}^\top\mathbf{H}\mathbf{B} + \frac{\lambda_2}{C}\mathbf{B} + \frac{\rho}{C}\mathbf{B} &= \mathbf{H}^\top\mathbf{T} + \frac{\rho}{C}(\mathbf{Z}^k - \mathbf{U}^k) \\
\left(\mathbf{H}^\top\mathbf{H} + \frac{\lambda_2 + \rho}{C}\mathbf{I}\right)\mathbf{B} &= \mathbf{H}^\top\mathbf{T} + \frac{\rho}{C}(\mathbf{Z}^k - \mathbf{U}^k) \\
\mathbf{B} &= \left(\mathbf{H}^\top\mathbf{H} + \frac{\lambda_2 + \rho}{C}\mathbf{I}\right)^{-1} \left(\mathbf{H}^\top\mathbf{T} + \frac{\rho}{C}(\mathbf{Z}^k - \mathbf{U}^k)\right) \\
\mathbf{B} &= \left(\mathbf{H}^\top\mathbf{H} + \eta\mathbf{I}\right)^{-1} \left(\mathbf{H}^\top\mathbf{T} + \frac{\rho}{C}(\mathbf{Z}^k - \mathbf{U}^k)\right)
\end{aligned}$$

onde $\eta = (\lambda_2 + \rho)/C$.

De forma similar, fazendo o gradiente da função objetivo, Equação 3.29, com respeito a \mathbf{B} igual a $\mathbf{0}$ tem-se,

$$\begin{aligned}
\rho\mathbf{H}^\top(\mathbf{H}\mathbf{B} - (\mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k)) + \lambda_2\mathbf{B} + (\mathbf{B} - \mathbf{Z}^k + \mathbf{U}_2^k) &= \mathbf{0} \\
\mathbf{H}^\top(\mathbf{H}\mathbf{B} - (\mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k)) + \frac{\lambda_2}{\rho}\mathbf{B} + (\mathbf{B} - \mathbf{Z}^k + \mathbf{U}_2^k) &= \mathbf{0} \\
\mathbf{H}^\top(\mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k) + (\mathbf{Z}^k - \mathbf{U}_2^k) &= \mathbf{H}^\top\mathbf{H}\mathbf{B} + \frac{\lambda_2}{\rho}\mathbf{B} + \mathbf{B} \\
\mathbf{H}^\top(\mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k) + (\mathbf{Z}^k - \mathbf{U}_2^k) &= \left(\mathbf{H}^\top\mathbf{H} + \frac{\lambda_2 + \rho}{\rho}\right)\mathbf{B}
\end{aligned}$$

Logo,

$$\mathbf{B} = \left(\mathbf{H}^\top\mathbf{H} + \eta\mathbf{I}\right)^{-1} \left[\mathbf{H}^\top(\mathbf{T} + \mathbf{E}^k - \mathbf{U}_1^k) + (\mathbf{Z}^k - \mathbf{U}_2^k)\right], \quad (\text{C.4})$$

onde $\eta = \frac{\lambda_2 + \rho}{\rho}$.

APÊNDICE D – Fatoração de Cholesky

Toda matriz definida positiva $\mathbf{A} \in \mathbb{R}^{n \times n}$ pode ser fatorada como

$$\mathbf{A} = \mathbf{L}\mathbf{L}^\top, \quad (\text{D.1})$$

onde \mathbf{L} é uma matriz triangular inferior com elementos da diagonal positivo. Assim, particiona-se a matriz $\mathbf{A} = \mathbf{L}\mathbf{L}^\top$ como

$$\begin{aligned} \begin{bmatrix} a_{11} & \mathbf{A}_{21}^\top \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} &= \begin{bmatrix} l_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} l_{11} & \mathbf{L}_{21}^\top \\ \mathbf{0} & \mathbf{L}_{22}^\top \end{bmatrix} \\ &= \begin{bmatrix} l_{11}^2 & l_{11}\mathbf{L}_{21}^\top \\ l_{11}\mathbf{L}_{21} & \mathbf{L}_{21}\mathbf{L}_{21}^\top + \mathbf{L}_{22}\mathbf{L}_{22}^\top \end{bmatrix}. \end{aligned} \quad (\text{D.2})$$

O algoritmo é composto por

- determinar l_{11} e \mathbf{L}_{21} :

$$l_{11} = \sqrt{a_{11}}, \quad \mathbf{L}_{21} = \frac{1}{l_{11}}\mathbf{A}_{21} \quad (\text{D.3})$$

- determinar \mathbf{L}_{22} a partir de

$$\mathbf{L}_{22}\mathbf{L}_{22}^\top = \mathbf{A}_{22} - \mathbf{L}_{21}\mathbf{L}_{21}^\top \quad (\text{D.4})$$

que é uma fatoração de Cholesky de ordem $n - 1$.

APÊNDICE E – Substituição *Forward* e *Backward*

Para solucionar o problema

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad (\text{E.1})$$

sendo \mathbf{A} uma matriz definida positiva de ordem n , utilizando substituição *forward* e *backward*, realiza-se os seguintes passos:

- fatorar \mathbf{A} como $\mathbf{A} = \mathbf{L}\mathbf{L}^\top$
- resolver $\mathbf{L}\mathbf{L}^\top \mathbf{x} = \mathbf{b}$
 - substituição *forward* $\mathbf{L}\mathbf{z} = \mathbf{b}$
 - substituição *backward* $\mathbf{L}^\top \mathbf{x} = \mathbf{z}$

A substituição *forward* para resolver $\mathbf{L}\mathbf{z} = \mathbf{b}$

$$\begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \quad (\text{E.2})$$

consiste nos seguintes passos

$$\begin{aligned} z_1 &= b_1/a_{11} \\ z_2 &= (b_2 - a_{21}z_1)/a_{22} \\ z_3 &= (b_3 - a_{31}z_1 - a_{32}z_2)/a_{33} \\ &\vdots \\ z_n &= (b_n - a_{n1}z_1 - a_{n2}z_2 - \cdots - a_{n(n-1)}z_{n-1})/a_{nn} \end{aligned}$$

Para substituição *backward* a ideia é a mesma, contudo, a matriz \mathbf{L}^\top é triangular superior.

APÊNDICE F – Normas

Norma de vetores é uma generalização de tamanho ou magnitude. Qualquer função real de \mathbf{x} , denotado por $\|\mathbf{x}\|$, pode ser definida como norma se atender as seguintes propriedades (CHEN, 1999):

- 1) $\|\mathbf{x}\| \geq 0$ para todo \mathbf{x} e $\|\mathbf{x}\| = 0$ se e somente se $\mathbf{x} = \mathbf{0}$;
- 2) $\|\alpha\mathbf{x}\| = |\alpha| \|\mathbf{x}\|$, para qualquer α real; e
- 3) $\|\mathbf{x}_1 + \mathbf{x}_2\| \leq \|\mathbf{x}_1\| + \|\mathbf{x}_2\|$ para todo \mathbf{x}_1 e \mathbf{x}_2 (desigualdade triangular).

Seja $\mathbf{x} = (x_1, x_2, \dots, x_n)^\top$. Então alguns exemplos da norma de \mathbf{x} são

$$\begin{aligned}\|\mathbf{x}\|_1 &:= \sum_{i=1}^n |x_i| \\ \|\mathbf{x}\|_2 &:= \sqrt{\mathbf{x}^\top \mathbf{x}} = \left(\sum_{i=1}^n |x_i|^2 \right)^{1/2} \\ \|\mathbf{x}\|_\infty &:= \max_i |x_i|.\end{aligned}$$

Estas são chamadas, respectivamente, norma- ℓ_1 , norma- ℓ_2 ou norma euclidiana, e norma infinita.

De maneira geral, define-se norma- ℓ_p como

$$\|\mathbf{x}\|_p := \left(\sum_{i=1}^n |x_i|^p \right)^{1/p}, \quad (\text{F.1})$$

com $0 < p \leq \infty$.

Seja $\mathbf{A} \in \mathbb{R}^{n \times m}$ uma matriz qualquer. A norma da matriz \mathbf{A} é um número não negativo associado à matriz, satisfazendo as seguintes propriedades (MEYER, 2000):

- 1) $\|\mathbf{A}\| \geq 0$ para todo \mathbf{A} e $\|\mathbf{A}\| = 0$ se e somente se $\mathbf{A} = \mathbf{0}$;
- 2) $\|\alpha\mathbf{A}\| = |\alpha| \|\mathbf{A}\|$, para qualquer α real; e
- 3) $\|\mathbf{A}_1 + \mathbf{A}_2\| \leq \|\mathbf{A}_1\| + \|\mathbf{A}_2\|$ para todo \mathbf{A}_1 e \mathbf{A}_2 (desigualdade triangular).

Suponha $\mathbf{A} = (\mathbf{a}_{1,\cdot}^\top, \mathbf{a}_{2,\cdot}^\top, \dots, \mathbf{a}_{n,\cdot}^\top)^\top$, onde $\mathbf{a}_{i,\cdot}$ representa a i -ésima linha de \mathbf{A} . Então alguns exemplos da norma de \mathbf{A} são

$$\begin{aligned}\|\mathbf{A}\|_F &:= \sqrt{\sum_{i=1}^n \sum_{j=1}^m |a_{ij}|^2} = \sqrt{\text{tr}(\mathbf{A}^\top \mathbf{A})} \\ \|\mathbf{A}\|_{2,1} &:= \sum_{i=1}^n \|\mathbf{a}_{i,\cdot}\|_2 = \sum_{i=1}^n \left(\sum_{j=1}^m |a_{ij}|^2 \right)^{1/2} \\ \|\mathbf{A}\|_* &:= \sum_{i=1}^{\min(n,m)} \sigma_i(\mathbf{A}),\end{aligned}$$

onde $\sigma_i(\mathbf{A})$ representa o valor singular de \mathbf{A} . Estas são chamadas, respectivamente, norma Frobenius, norma- $\ell_{2,1}$ e norma nuclear.

Note que a norma Frobenius é uma extensão da norma ℓ_2 para matrizes. Além disso, a norma $\ell_{2,1}$ é equivalente a norma ℓ_1 quando m é igual a um, isto é, quando tem-se um vetor ao invés de uma matriz.

APÊNDICE G – Testes Estatísticos

Para comparação de vários métodos sobre vários bancos de dados utiliza-se o teste não paramétrico de Friedman (FRIEDMAN, 1937). O teste ordena os métodos, para cada banco de dados, de acordo com o desempenho deles. Assim, o método com melhor resultado recebe *rank* 1, o segundo melhor *rank* 2 e assim por diante. No caso de empates a média dos *ranks* é feita e atribuída a todos os métodos empatados.

Considere k métodos e D bancos de dados. Seja R_j o *rank* médio do j -ésimo método sobre os D bancos de dados. O teste de Friedman, possui como hipótese nula que todos os métodos são equivalentes e, portanto, os *ranks* médios R_j , devem ser iguais. A estatística de teste de Friedman é dada por

$$\chi_F^2 = \frac{12D}{k(k+1)} \left[\sum_{j=1}^k R_j^2 - \frac{k(k+1)^2}{4} \right], \quad (\text{G.1})$$

e possui distribuição $\chi_{(k-1)}^2$, quando D e k são grandes o suficiente ($D > 10$ e $k > 5$). Quando k e D são pequenos, a distribuição exata deve ser calculada (ZAR, 1998; SHESKIN, 2000).

Iman e Davenport (1980) mostraram que a estatística de Friedman é muito conservadora e propuseram a seguinte estatística

$$F_F = \frac{(D-1)\chi_F^2}{D(k-1) - \chi_F^2}, \quad (\text{G.2})$$

que possui distribuição $F_{(k-1), (k-1)(D-1)}$.

Caso a hipótese nula seja rejeitada, isto é, os métodos não são equivalentes, um segundo teste é realizado para comparar os métodos entre si. O segundo teste pode ser o teste de Nemenyi (NEMENYI, 1963). O desempenho de dois métodos será significativamente diferente se a diferença entre os *ranks* médios dos dois métodos for maior que a distância crítica (DC)

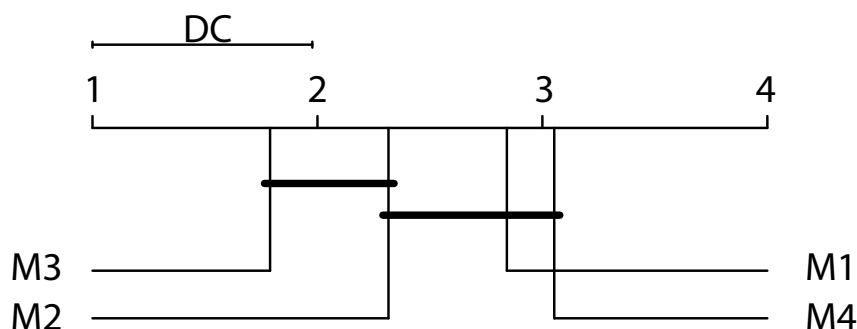
$$DC = q_\alpha \sqrt{\frac{k(k+1)}{6D}}, \quad (\text{G.3})$$

onde q_α é o valor crítico, para um nível de significância α , baseado na estatística de amplitude estudentizada dividida por $\sqrt{2}$, valor tabelado disponibilizado em (DEMŠAR, 2006). Um nível de significância de 5% foi usado para ambos os testes nos experimentos desta Tese.

Os pressupostos para aplicação do teste de comparação de métodos sobre vários bancos de dados são: os resultados medidos devem ser confiáveis; todos os métodos avaliados sobre as mesmas partições; e uma quantidade mínima de bancos de dados seja usada no teste.

Os resultados dos testes estatísticos podem ser representados por um diagrama simples (DEMŠAR, 2006). A Figura G.1 exibe um exemplo de resultado para quatro métodos, M1, M2, M3 e M4. A linha superior do diagrama é o eixo no qual é indicado o *rank* dos métodos. O eixo é orientado no sentido dos menores (melhores) *rank* estarem do lado esquerdo.

Figura G.1 – Comparação de quatro métodos entre si através com o teste de Nemenyi. Grupos de métodos que não são estatisticamente diferentes (a um nível de significância de 0.05) estão conectados por uma linha espessa.



Fonte: Produção do próprio autor.

Os grupos de métodos que não são estatisticamente diferentes são conectados por uma linha espessa, e a distância crítica (DC) entre dois métodos, para serem considerados estatisticamente diferentes, é indicada acima do gráfico. Assim, analisando o diagrama da Figura G.1, segue que os métodos M3 e M2 não são estatisticamente diferentes. Além disso, os métodos M1, M2 e M4 não apresentam diferença significativa. Por fim, o método M3 apresentou-se estatisticamente superior aos métodos M1 e M4.

Até o momento, foram abordados os testes estatísticos aplicados quando deseja-se comparar vários métodos sob vários bancos de dados. Para o caso onde deseja-se comparar apenas dois métodos em várias bases de dados, o teste *Wilcoxon Signed Rank* (WILCOXON, 1945) pode ser utilizado.

Esse teste ordena as diferenças de desempenho de dois métodos para cada banco de dados, onde os sinais dessas diferenças são ignorados, e compara com o *rank* para as diferenças positivas e negativas.

Seja d_i a diferença entre a medida de desempenho entre dois métodos na i -ésima base de dados (de um total de D conjuntos). As diferenças são ordenadas de acordo com o valor absoluto de d_i e a média é atribuída aos casos de empate. Seja R^+ a soma dos *ranks* das bases de dados que o segundo método superou o primeiro, e R^- a soma dos *ranks* das bases de dados onde o primeiro método superou o segundo. Os *ranks* associados ao $d_i = 0$ são divididos de forma iguais entre as somas. Havendo um número ímpar de $d_i = 0$, uma

amostra é ignorada. Assim,

$$R^+ = \sum_{d_i > 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i), \quad R^- = \sum_{d_i < 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i). \quad (\text{G.4})$$

Seja $W = \min(R^+, R^-)$. Os valores críticos exatos para W e N até 25 são tabelados e encontram-se na maioria dos livros de estatística (DEMŠAR, 2006). Para um número maior de conjunto de dados, a estatística

$$Z = \frac{W - \frac{1}{4}N(N+1)}{\sqrt{\frac{1}{24}N(N+1)(2N+1)}}, \quad (\text{G.5})$$

possui, aproximadamente, distribuição normal.